1998

Vision guidance controller for an Unmanned Aerial Vehicle

Watson, Mark T.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

VISION GUIDANCE CONTROLLER FOR AN UNMANNED AERIAL VEHICLE

By

Mark T. Watson

December 1998

Thesis Advisor: Issac I. Kaminer

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1998	Master's Thesis

4. TITLE AND SUBTITLE
VISION GUIDANCE CONTROLLER FOR AN UNMANNED AERIAL VEHICLE

5. FUNDING NUMBERS

6. AUTHOR(S)
Watson, Mark T.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The use of Unmanned Aerial Vehicles (UAVs) in modern military operations for reconnaissance and other missions continues to grow. UAV systems using remote control guidance are limited in range and subject to Electronic Warfare concerns. Guidance systems using only Global Positioning Service (GPS) or an Inertial Navigation System (INS) are limited to a pre-programmed route of flight. A vision guidance system that can control the UAV over an arbitrary course is not subject to these limitations. This thesis uses classical control methods to develop and test an autonomous vision controller for the FOG-R UAV (FROG). First, a computer model of the camera output for a flight that tracks a river is made to develop the controller and to test it in nonlinear simulation. Finally, the complete system is flight tested on the FROG UAV.

The design and test equipment include a highly modified FOG-R UAV from the U.S. Army, the MATRIX$_X$ Product Family of software tools developed by Integrated Systems, Inc., and a Ground Station built at NPS from commercially available computer and communication equipment.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Unmanned Aerial Vehicles, FROG, Rapid Prototyping, Vision Guidance	94
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFI- CATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited

VISION GUIDANCE CONTROLLER FOR AN UNMANNED AERIAL VEHICLE

Mark T. Watson
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1998

Author: [redacted]
Mark T. Watson

Approved by: [redacted]
Isaac I. Kaminer, Thesis Advisor

[redacted]
Russell W. Duren, Second Reader

[redacted]
Gerald H. Lindsey, Chairman
Department of Aeronautics and Astronautics

ABSTRACT

The use of Unmanned Aerial Vehicles (UAVs) in modern military operations for reconnaissance and other missions continues to grow. UAV systems using remote control guidance are limited in range and subject to Electronic Warfare concerns. Guidance systems using only Global Positioning Service (GPS) or an Inertial Navigation System (INS) are limited to a pre-programmed route of flight. A vision guidance system that can control the UAV over an arbitrary course is not subject to these limitations. This thesis uses classical control methods to develop and test an autonomous vision controller for the FOG-R UAV (FROG). First, a computer model of the camera output for a flight that tracks a river is made to develop the controller and to test it in nonlinear simulation. Finally, the complete system is flight tested on the FROG UAV.

The design and test equipment include a highly modified FOG-R UAV from the U.S. Army, the MATRIX$_X$ Product Family of software tools developed by Integrated Systems, Inc., and a Ground Station built at NPS from commercially available computer and communication equipment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENT

I would like to thank my advisor, Dr. Isaac Kaminer, for his guidance and patience in the completion of this endeavor. I would also like to thank Don Meeks and Jerry Lentz without whom none of the flight tests would have taken place. Lastly, I would especially like to thank my wife, Mary, who supported me (and put up with me) during this process.

I. INTRODUCTION

A. PURPOSE

The primary purpose of this thesis was to design, simulate on the ground and flight test an autonomous vision guidance system for an Unmanned Aerial Vehicle (UAV). To achieve this, a model of the vision sensor was also developed. The vision guidance system controls the UAV in the horizontal plane to follow an arbitrary curve on the ground. The guidance control system was designed to take the output from a digital image processor to track the arbitrary curve. This processor uses the image obtained by an onboard video camera in the nose of the aircraft. The image processor uses color to distinguish a contiguous curve, such as a river or road, in each picture frame sent to it from a digital frame grabber. The processor then takes the pixel coordinates of this curve in the camera image plane that corresponds to a pre-determined distance ahead of the UAV and sends it to the controller. A computer model for the camera and processor sensor unit was needed to develop the controller and to test it during ground simulations. Additionally, the digital image processor programming and hardware interfaces are still under development necessitating the use of the camera model during the actual flight testing of the controller.

The secondary purpose of this thesis was to implement and flight test improvements to the existing Flight Management System (FMS) for the Naval Postgraduate School's UAV. The FMS has been evolving as previous thesis students have developed individual controllers for altitude, speed, and heading. Integrating these different controllers for combined operation during flight testing was needed to improve safety of the test vehicle

and to simplify use for the operator. Changes to the FMS include coding changes to the user defined blocks of the existing FMS software, changes to the wind down filters, and the addition of switches to choose between absolute or delta commands.

B. HARDWARE

Only a brief description of the hardware involved is required for the presentation of this project; however, a comprehensive description can be found in Froncillo, Komlosy, and Rivers [Refs. 1-3].

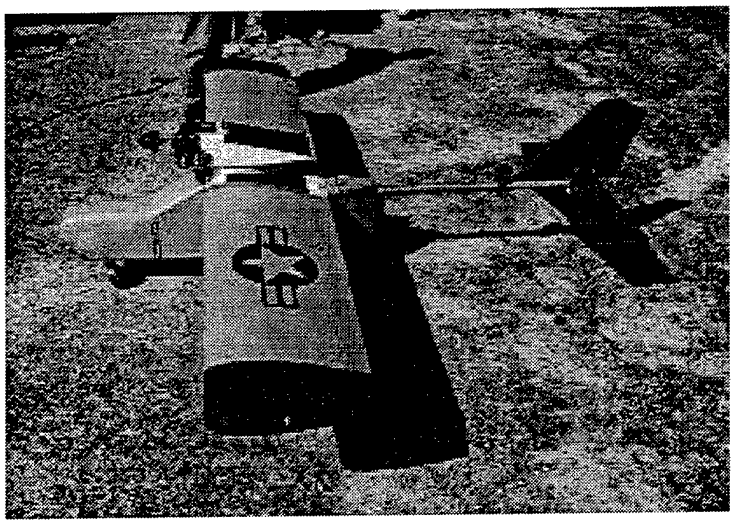

Figure 1 FROG UAV

1. The Airborne Components

The flight vehicle used in the design and testing of this controller was the U.S. Army's FOG-R UAV. Nicknamed the FROG (Figure 1), it is a high wing monoplane with the engine mounted on a pylon above the twelve foot wingspan. Takeoff weight is typically 90 lbs., including 20 lbs. of payload used for the sensor package. The FROG uses conventional controls of ailerons, elevator, and rudder. The aircraft is Radio Controlled

(RC) via Futaba Pulse Width Modulation (PWM) transmitters of the type used by sport RC modelers. The FROG also has an onboard autopilot with its own yaw and climb rate sensors. This allows the pilot to fly the aircraft via direct control surface movement or by commanding turn and climb rates through the autopilot. The same control options apply to the FMS.

The sensor package consists of a pitot-static system for airspeed and altitude, five single turn potentiometers for the positions of control surfaces and relative wind angles, a differential Global Positioning Service (GPS) unit, and an Inertial Measurement Unit (IMU). The IMU measures body angles, angle rates, accelerations, and the earth's magnetic field for all three axes of the aircraft. The IMU also has five Analog to Digital (A/D) converters that are used to send velocity and four other parameters to the ground station. The parameters from the potentiometers and pitot-static system that are available for transmission are angle of attack, side slip, the three control surfaces position, dynamic pressure, and static pressure. For this project only dynamic pressure, static pressure, and side slip sensors were connected to the IMU. All data from the IMU and the differential GPS is transmitted to the ground station by two spread spectrum radio frequency (RF) modems at a 9600 Baud data rate. The GPS modem operates in the full duplex mode receiving differential corrections from the ground station's GPS receiver. Figure 2 shows the hardware layout and interfaces.

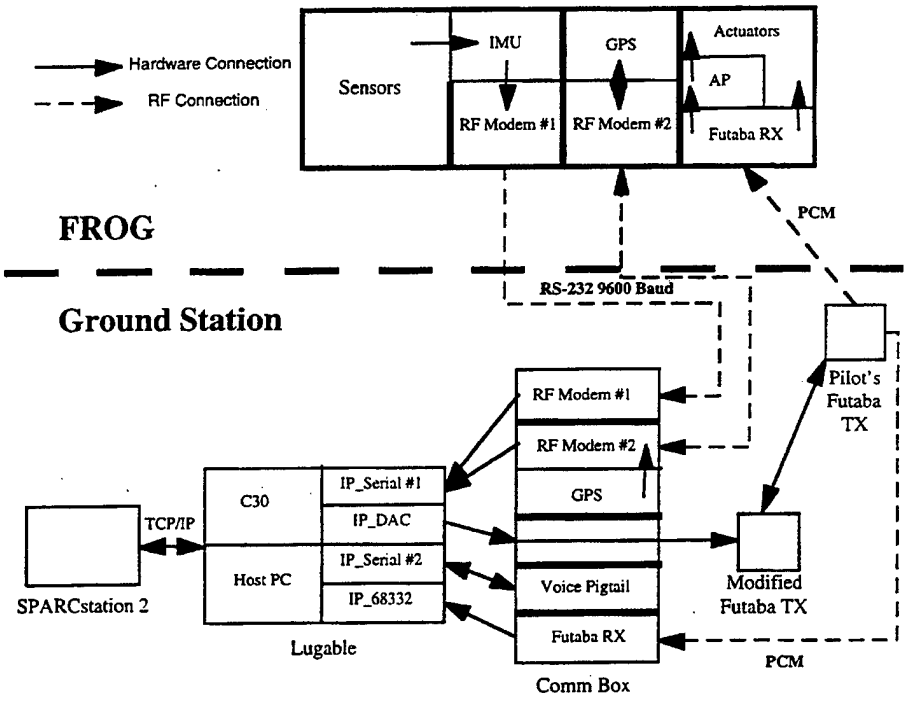

Figure 2 Hardware Interfaces

2. The Ground Components

The ground station is responsible for flight control of the UAV and data collection. As seen in Figure 2, the ground station consists of three major components. The SPARC 2 workstation executes all the software used for initial development, ground simulation, and control of the aircraft during flight test. All the data storage and analysis is also performed on the SPARC 2. The luggable computer contains the host processor and the AC-100 Model C30 real-time hardware controller. The host computer handles File Transfer Protocol (FTP), compile, link, and download functions of the system. The luggable also has four IP modules that provide the interface with the communication box as shown in Figure 2. The communications box with the associated antennas has all the equipment to transmit and receive data and control commands between the ground station and the UAV. This

includes two RF modems, a GPS receiver, and a Futaba PWM receiver identical to the ones in the aircraft. The FROG is controlled using two Futaba RC controllers. One controller, referred to as the "slave", has been modified to accept inputs from the computer via the IP_DAC module. The pilot uses a standard Futaba controller as the "master" controller. When the pilot holds down the trainer switch on the "master", control is passed to the "slave" and, hence, the computer.

C. RAPID PROTOTYPING

As with the description of the hardware components, the use of RealSim for the design of control systems was explored in earlier thesis projects; consequently, only a brief summary of the rapid prototyping system is provided below. A more complete description can be found in Froncillo, Komlosy, and Rivers [Refs. 1-3].

A rapid prototyping system allows the engineer to quickly design, test, and implement a control system. The MATRIX$_X$ Product Family of software tools is a commercial system that provides a set of integrated tools to accomplish this task. The functionality of each MATRIX$_X$ tool is shown in Figure 3. Xmath/SystemBuild is an interrelated software package similar to MATLAB/Simulink. The RealSim Graphical User Interface (GUI), shown in Figure 4, provides overall control by stepping the user through the design process from initial formulation to the actual real-time implementation of the control system.

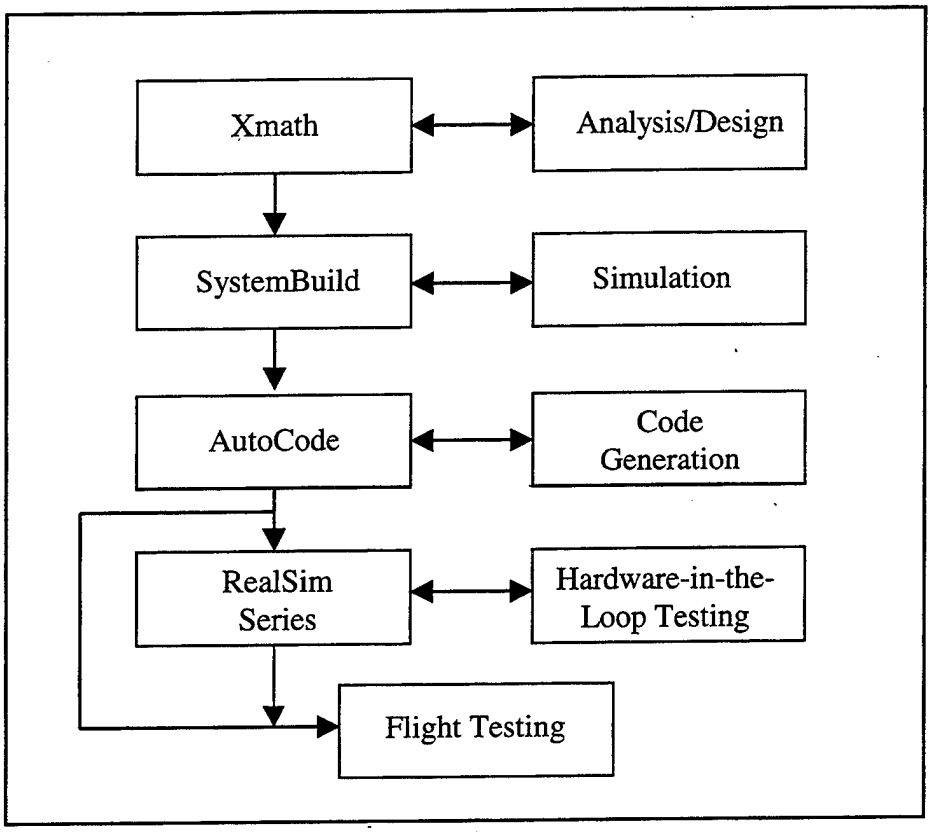

Figure 3 RealSim Functions

Xmath is the computational element that provides analysis and control simulation functions. SystemBuild is a graphical, interactive program that uses both pre-defined and user defined blocks to model system elements. The autocode feature is a powerful time saving capability that generates high level C code based on the system built in the graphical System/Build environment. Once the code is generated it is sent to the host computer via FTP. The host computer compiler generates the object code and the link produces the executable code for the target processor. The animation builder enables the user to build a graphical interface with the control system that allows real-time inputs as well as display of

system outputs during both ground and flight testing. The hardware connection editor is used to associate system inputs and outputs with external hardware. The final feature of the RealSim GUI is the download and run feature, which loads the executable code into the target processor and initiates the real-time operation.

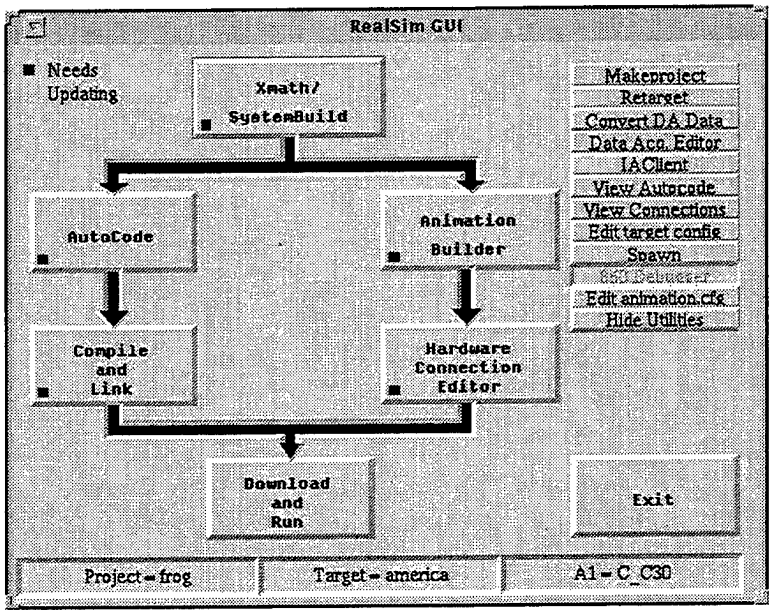

Figure 4 RealSim GUI

II. FLIGHT MANAGEMENT SYSTEM

The FMS is used to control the FROG during flight tests. The animation page, shown in Figure 5, allows the user to monitor all the critical flight parameters and to command inputs via the different controllers. The FMS has been modified to incorporate an Absolute or a Delta mode of operation for both the altitude and the heading controllers. All commands entered from the FMS animation page are converted to analog voltages that are sent to the Futaba controller. They are then converted to PWM signals and transmitted to the FROG. The interfaces are described in the hardware section of Chapter I. It should be noted that none of the performance characteristics of the different controllers has been altered, and therefore no analysis of the performance is done. The changes only affect the implementation of the controllers and were made to enhance safety of the vehicle and to ease operation by the user. Flight test and ground simulations were made only to test proper implementation.

A. ALTITUDE CONTROLLER

The altitude controller for the FROG was developed to send climb rate commands through the onboard autopilot [Ref. 1]. The original controller had two modes of operation: Open Loop (OL) or Closed Loop (CL). For the OL mode the operator enters the desired climb rate in feet per minute on the FMS page. No altitude or climb rate is referenced or fed back to the controller. The OL mode has not been changed and is still available to the operator.

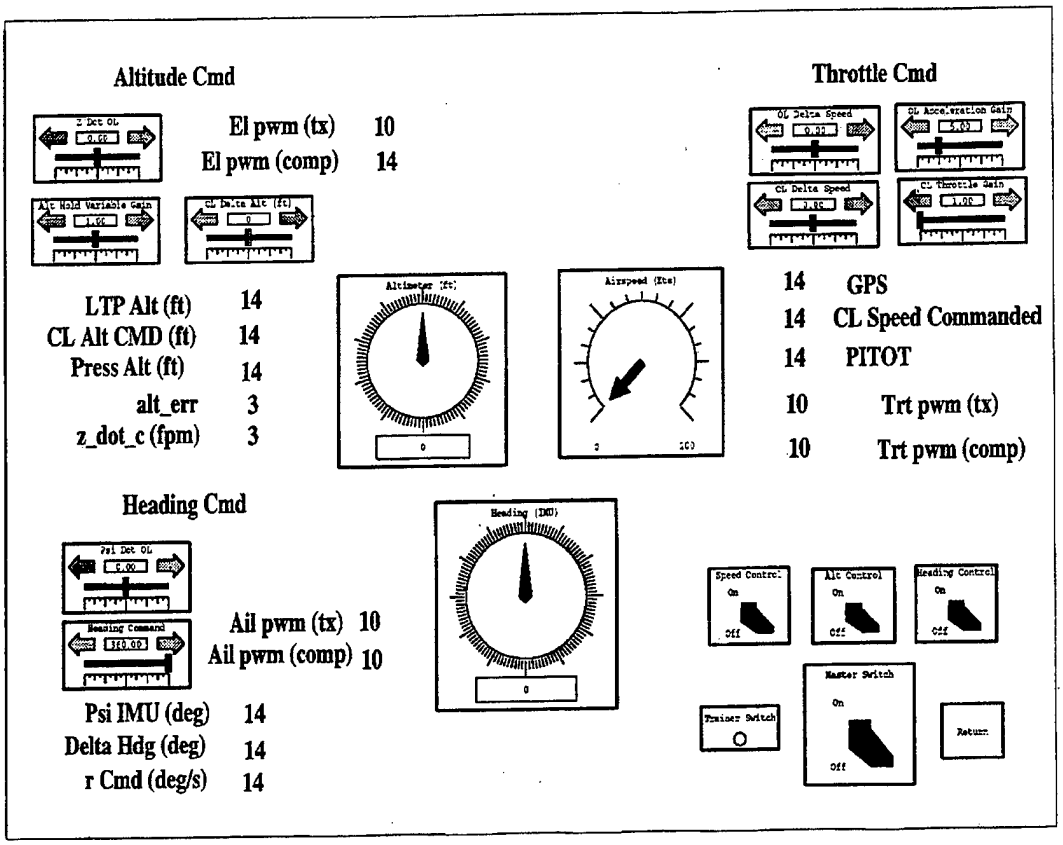

Figure 5 FMS Page

The original CL mode consisted only of a Delta altitude mode. The desired altitude change in feet was entered on the FMS page. The reference pressure altitude at the time the altitude CL switch was turned on was frozen and added to the altitude change commanded. This altitude sum was compared to the actual pressure altitude and fed back through a Proportional-plus-Integral-plus-Derivative (PID) controller that commanded the required climb rate to achieve the desired change in altitude. The addition of an absolute altitude mode and the Master switch requires a change to this logic. Now the reference altitude is not frozen until the Master switch is on, the altitude CL switch is on, and the altitude

Absolute/Delta switch is in the Delta position. This prevents the reference altitude from being set when operating in the absolute mode, and it allows the subordinate altitude CL switch to be set prior to the test run and activated via the Master switch. The reference altitude can be cleared and reset by cycling any of these three switches.

An absolute altitude mode was added that allows the operator to command an altitude based only on the pressure altimeter. The resulting altitude command is either an above ground level (agl) altitude if the altimeter is zeroed prior to flight, or a mean sea level (msl) altitude if the field elevation is set into the altimeter prior to flight. An altitude of 250 feet is set as the minimum possible commanded altitude as a safety feature. With a test field elevation of 80 feet, this provides adequate ground clearance for both agl and msl altimeter settings.

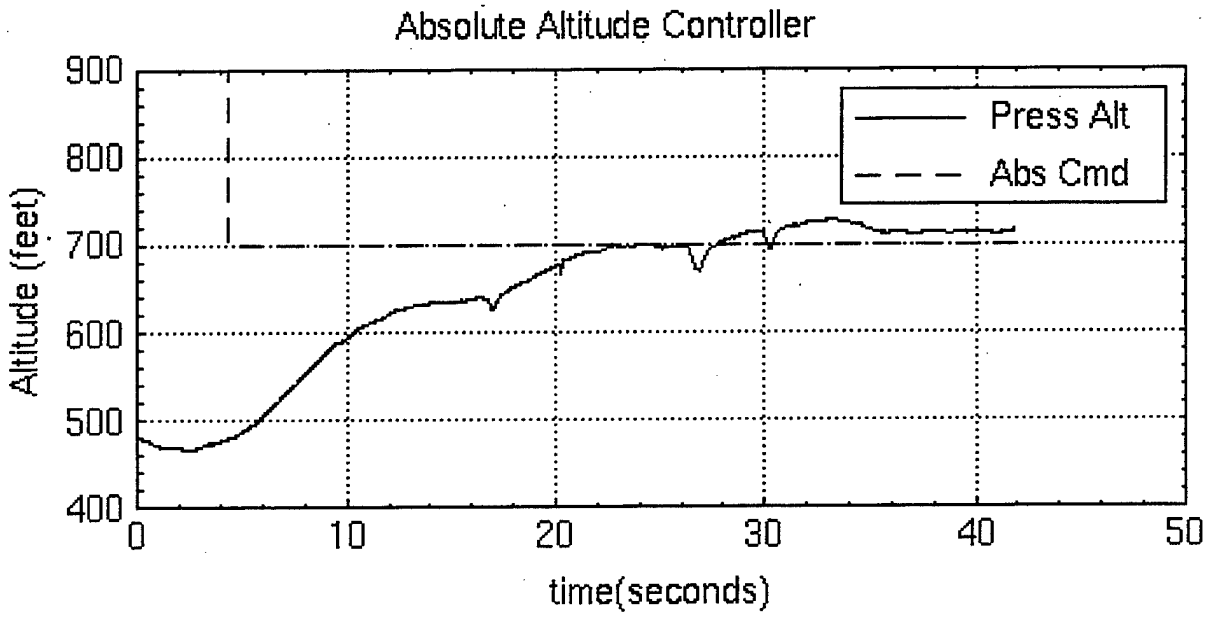

Figure 6 Absolute Altitude Test

Figure 6 shows the response during a flight test run of the absolute altitude mode. At the start of the run the FROG was at 480 feet agl and was immediately commanded to 700 feet agl. The data shows the FROG climbing and maintaining the desired altitude. The default mode at system initiation is the Delta mode with a change of zero feet commanded. This is a safety feature that minimizes the possibility of an abrupt altitude change when the computer is given control of the aircraft. Figure 7 shows the new altitude controller with the logic blocks added to implement these features.

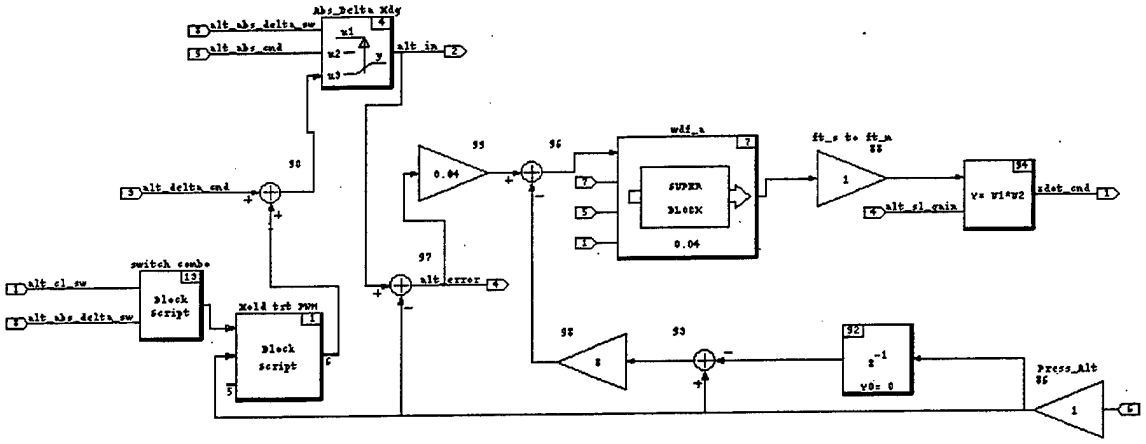

Figure 7 Altitude Controller

B. HEADING CONTROLLER

The heading controller for the FROG has an OL mode of operation similar to that described under the altitude controller section. It commands a yaw rate to the FROG autopilot directly from the value entered on the FMS page. No changes were made to the OL mode.

The original CL heading mode consisted only of an Absolute mode. The desired heading in degrees from 0 to 359 is entered on the FMS page and compared to the actual heading from the IMU. This is sent through a PID controller that calculates the commanded yaw rate required to achieve the desired heading [Ref. 3]. This mode is still available when the heading Absolute/Delta switch is in the Absolute position. A new Delta mode was added with the same logic as in the altitude controller. A reference heading is frozen when the Master switch is on, the heading CL switch is on, and the Absolute/Delta switch is in the Delta position. The reference heading can be cleared and reset by cycling any of these three switches. The default mode at system initiation is the Absolute mode with 360° commanded. With 360° entered the yaw rate commanded is zero. This is a safety feature that prevents abrupt commands when the controller is engaged and also allows a zero yaw rate command, or "steady up" signal, to be sent during CL operation.

Figure 8 shows the results of a test run for a Delta command of right 45°. When the CL switch was thrown at six seconds, the reference heading of 225° was frozen. The controller then turned the aircraft to 270° and maintained that heading. Figure 9 shows the same test to the left. At two seconds the reference heading of 240° was frozen. The controller turned the aircraft to 195° and maintained that heading. Figure 10 shows the new heading controller with the logic blocks added to implement these features.

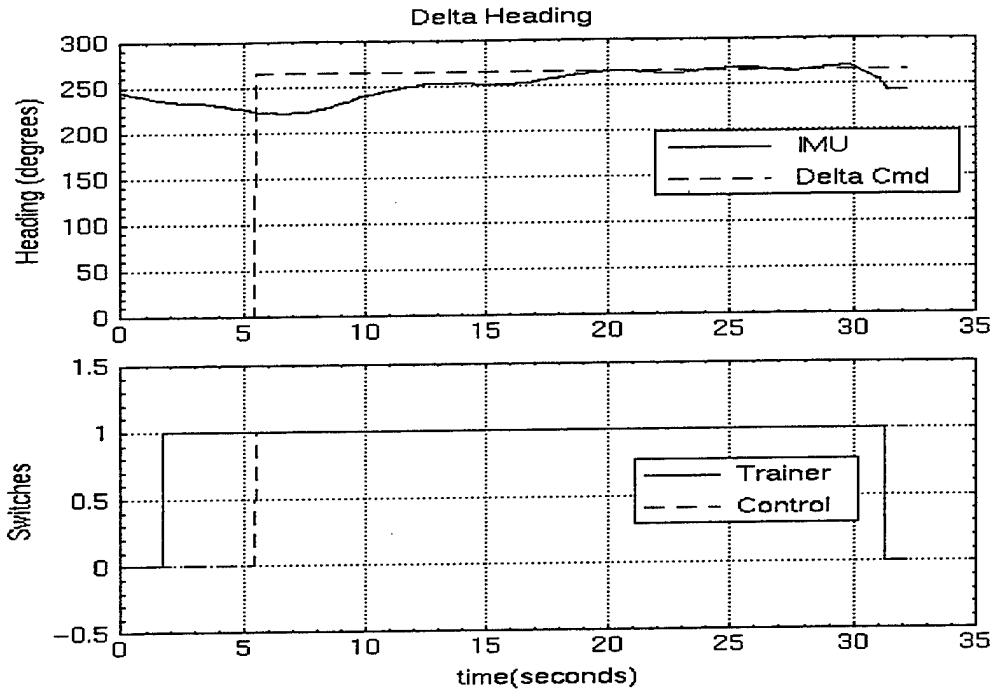

Figure 8 Right 45 Degrees

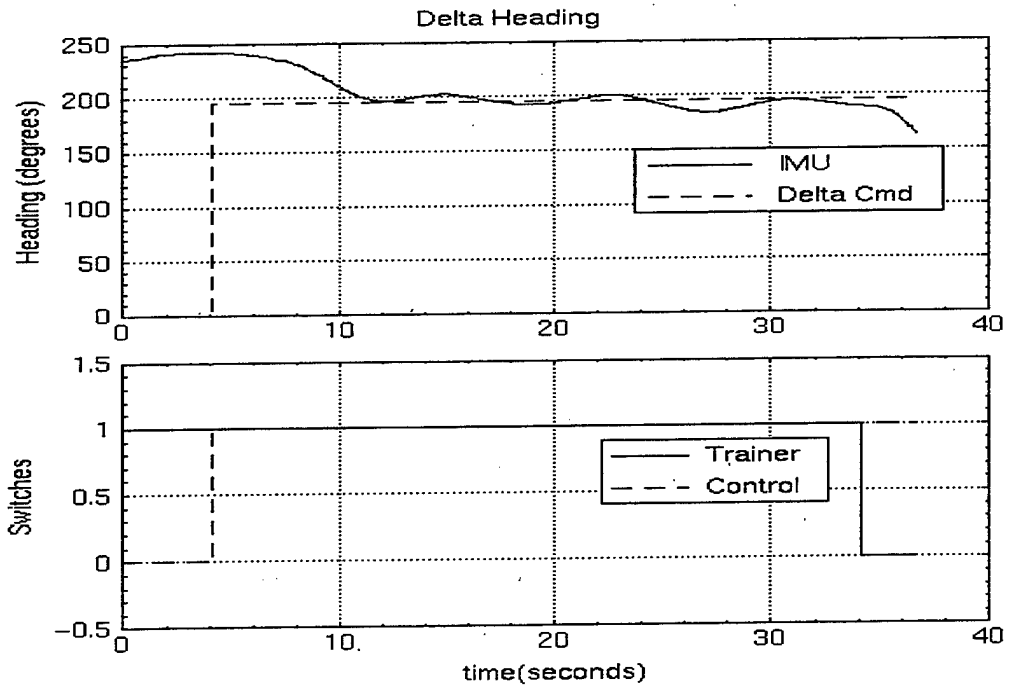

Figure 9 Left 45 Degrees

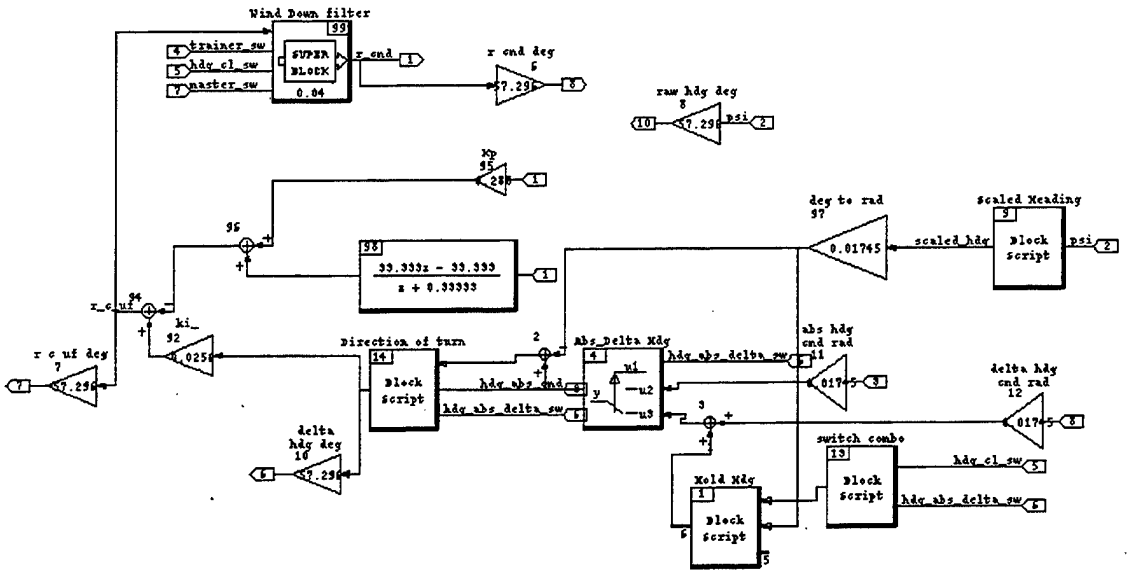

Figure 10 Heading Controller

C. WIND DOWN FILTERS

The Master switch has been added to each wind down filter to ensure proper operation. All CL control switches are now subordinate to the Master switch. This allows the individual controllers that are going to be tested during a run to be selected ahead of time. They are then turned on and off simultaneously via the Master switch. Figure 11 shows the wind down filter used in the heading controller. This configuration is the same as the altitude controller. The wind down loop incorporates all three switches connected to each controller: the Trainer switch, the Master switch, and the CL switch for each individual controller. The integrator forces the output back to its initial value if any of the

three switches are off. This prevents the control command from building up while the controller is not selected.

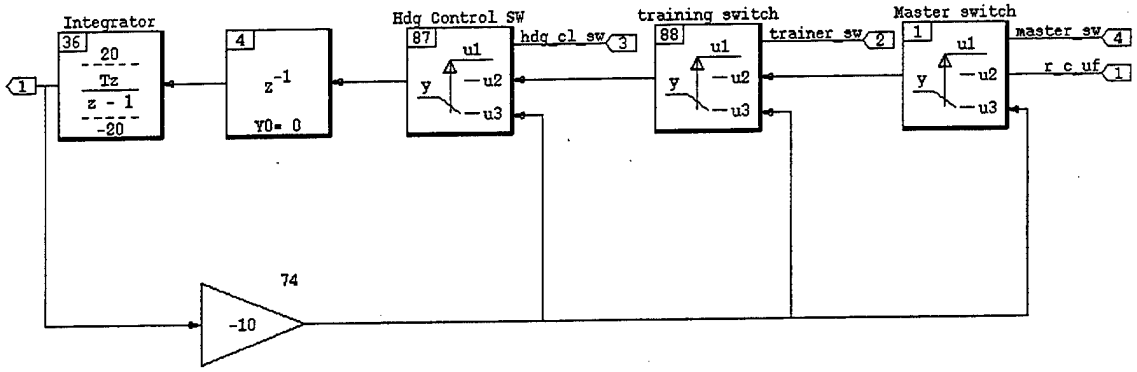

Figure 11 Winddown Filter

III. COORDINATE SYSTEMS

The development of the camera model and the vision controller requires an introduction of several different coordinate systems, including Earth Centered Earth Fixed (ECEF), Local Tangent Plane (LTP), Body, Camera, and Image plane systems. This chapter will address the relationships and the transformations between the various coordinate systems.

A. ECEF AND LTP {U}

Consider a Cartesian coordinate system with its origin at the center of the earth, oriented such that its z-axis is aligned due north, its x-axis intersects the prime meridian, and its y-axis completes the right hand rule. This coordinate system is called ECEF [Ref. 4]. LTP coordinate systems are defined by a plane that is tangent to a point on the surface of the earth. The point of tangency is defined as the origin of the LTP coordinate system. The x-axis is in the plane and points toward true north. The y-axis is in the plane perpendicular to the x-axis and points toward east. The z-axis is perpendicular to the plane. If the z-axis points away from the center of the earth as shown in Figure 12, it is referred to as a North-East-Up (NEU) orientation. If the positive z-axis points towards the center of the earth it is referred to as a North-East-Down (NED) orientation. The NED orientation will be used since it is a right hand system and easily corresponds with the standard aircraft Body system to be defined later.

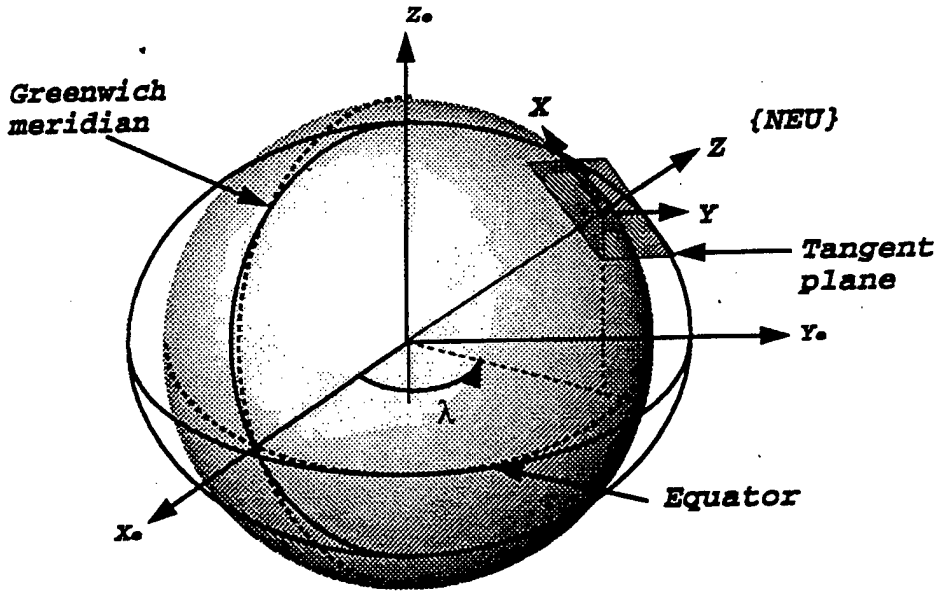

Figure 12 ECEF and LTP

Since both the aircraft's position given by the GPS and the location of the LTP origin are specified by latitude and longitude, it is necessary to convert these to ECEF coordinates. Let $[\lambda_1, \lambda_2, h]^T$ be the position of any arbitrary point, where

λ_1 = degrees of latitude,

λ_2 = degrees of longitude,

and

h = height in meters.

The height of the point is measured above the surface of the earth. GPS uses an ellipsoidal model for the earth based on the World Geodetic Survey of 1984 (WGS84) [Ref. 4]. The

two parameters that describe an ellipsoid are its eccentricity (ε) and the length of its semi-major axis (a). For the WGS84 ellipsoid these values are $\varepsilon = 0.00669437999013$ and a = 6,378,137 meters. With these values the distance from the center of the WGS84 ellipsoid to a point on its surface, where the local latitude is λ_1, is given by

$$N = \frac{a}{\sqrt{1-\varepsilon^2 \sin^2(\lambda_1)}}, \tag{III.1}$$

and the ECEF coordinates of the arbitrary point are

$$\begin{aligned} x &= (N+h)\cos(\lambda_2)\cos(\lambda_1). \\ y &= (N+h)\cos(\lambda_2)\sin(\lambda_1). \\ z &= \left(N(1-\varepsilon^2)+h\right)\sin(\lambda_2). \end{aligned} \tag{III.2}$$

Since all calculations in the sensor model are done in LTP coordinates, the ECEF coordinates are transformed to LTP.

Let

$[\lambda_{1o}, \lambda_{2o}, h_o]^T$ = the surveyed position of the origin of the LTP,

$\mathbf{P}_P$ = the position of the point in ECEF coordinates,

$\mathbf{P}_O$ = the position of the LTP origin in ECEF coordinates,

$\mathbf{P}_{P\_LTP}$ = the position of the point relative to the LTP origin resolved in ECEF,

${}^{LTP}_{ECEF}R$ = the rotation matrix from ECEF to a NED LTP

$$\begin{bmatrix} -\sin(\lambda_{2o})\cos(\lambda_{1o}) & -\sin(\lambda_{2o})\sin(\lambda_{1o}) & \cos(\lambda_{2o}) \\ -\sin(\lambda_{1o}) & \cos(\lambda_{1o}) & 0 \\ -\cos(\lambda_{2o})\cos(\lambda_{1o}) & -\cos(\lambda_{2o})\sin(\lambda_{1o}) & -\sin(\lambda_{2o}) \end{bmatrix}, \tag{III.3}$$

${}^{LTP}\mathbf{P}_P$ = the position of the point in LTP coordinates.

Then,

$$\mathbf{P}_{P\_LTP} = \mathbf{P}_P - \mathbf{P}_O, \tag{III.4}$$

and

$$^{LTP}\mathbf{P}_P = {}^{LTP}_{ECEF}R \ \mathbf{P}_{P\_LTP}. \tag{III.5}$$

B. BODY REFERENCE FRAME {B}

The aircraft Body reference frame is a right hand orthogonal coordinate system with the origin at the aircraft's center of gravity. The x-axis points forward along the longitudinal axis. The y-axis points out toward the right wing, and the z-axis points down completing the right hand rule. Figure 13 shows the Body reference frame.

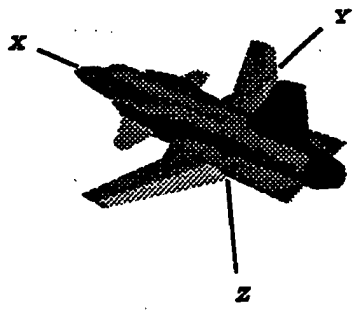

Figure 13 Body Reference Frame

Euler angles are used to define the orientation of two coordinate systems with respect to each other. These angles define how the Body reference frame fixed to the aircraft is oriented with respect to the inertial LTP reference frame. The angles shown in Figure 14 are ϕ for roll, θ for pitch , and ψ for yaw.

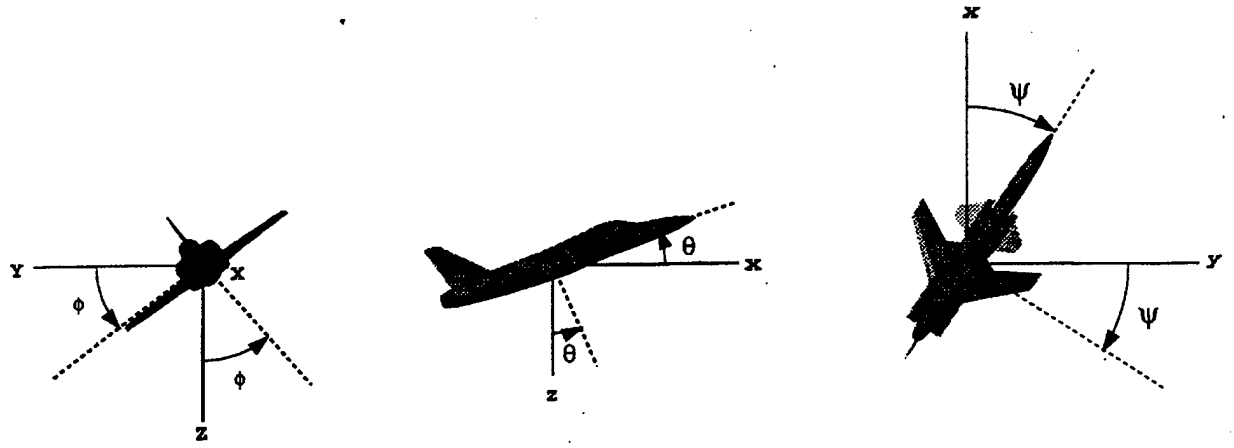

Figure 14 Euler Angles

There are many possible combinations of rotation matrices. The rotation matrix used with conventional aircraft to transform from {U} to {B} is the 3-2-1 rotation matrix that rotates in yaw first, then pitch, and then roll.

$$ {}^{B}_{U}R = \begin{bmatrix} \cos(\psi)\cos(\theta) & \sin(\psi)\cos(\theta) & -\sin(\theta) \\ \cos(\psi)\sin(\theta)\sin(\phi)-\sin(\psi)\cos(\phi) & \sin(\theta)\sin(\phi)\sin(\psi)+\cos(\psi)\cos(\phi) & \cos(\theta)\sin(\phi) \\ \cos(\psi)\sin(\theta)\cos(\phi)+\sin(\psi)\sin(\phi) & \sin(\theta)\cos(\phi)\sin(\psi)-\cos(\psi)\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \quad \text{(III.6)} $$

The inverse of this matrix is used to convert from {B} to {U}.

$$ {}^{U}_{B}R = \begin{bmatrix} \cos(\theta)\cos(\psi) & -\cos(\phi)\sin(\psi)+\sin(\phi)\sin(\theta)\cos(\psi) & \sin(\phi)\sin(\psi)+\cos(\phi)\sin(\theta)\cos(\psi) \\ \cos(\theta)\sin(\psi) & \cos(\phi)\cos(\psi)+\sin(\phi)\sin(\theta)\sin(\psi) & -\sin(\phi)\cos(\psi)+\cos(\phi)\sin(\theta)\sin(\psi) \\ -\sin(\theta) & \sin(\phi)\cos(\theta) & \cos(\phi)\cos(\theta) \end{bmatrix} \quad \text{(III.7)} $$

C. CAMERA REFERENCE FRAME {C}

The Camera reference frame is a Cartesian coordinate system with its origin located at the focal point of the camera. The x-axis points forward along the optical axis of the camera. The y-axis is positive to the right, and the z-axis is positive down as shown in Figure 15. With the camera mounted in the aircraft, the conversion from {B} to {C} would use the same 3-2-1 rotation matrix, ${}^{C}_{B}R$, as Eq. III.6. If the camera is mounted coincident with the aircraft body axes, then all the Euler angles are zero, the rotation matrix is the identity matrix, and {C} would equal {B}. For this application the only non-zero Euler angle is θ_C that accounts for the depression angle of the nose mounted camera.

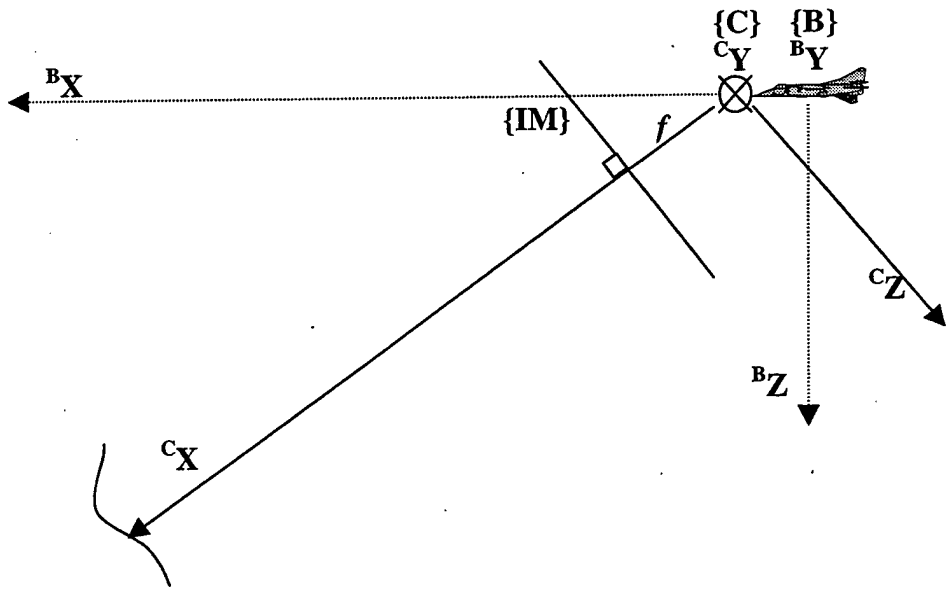

Figure 15 Camera Reference Frame

D. IMAGE PLANE {IM}

To model the camera required transforming the three dimensional reference system {C} to a two dimensional camera Image plane reference system. A pinhole camera model as shown in Figure 16 and discussed in Kaminer [Ref. 4] was used to accomplish this mapping of three dimensional coordinates to a two dimensional system.

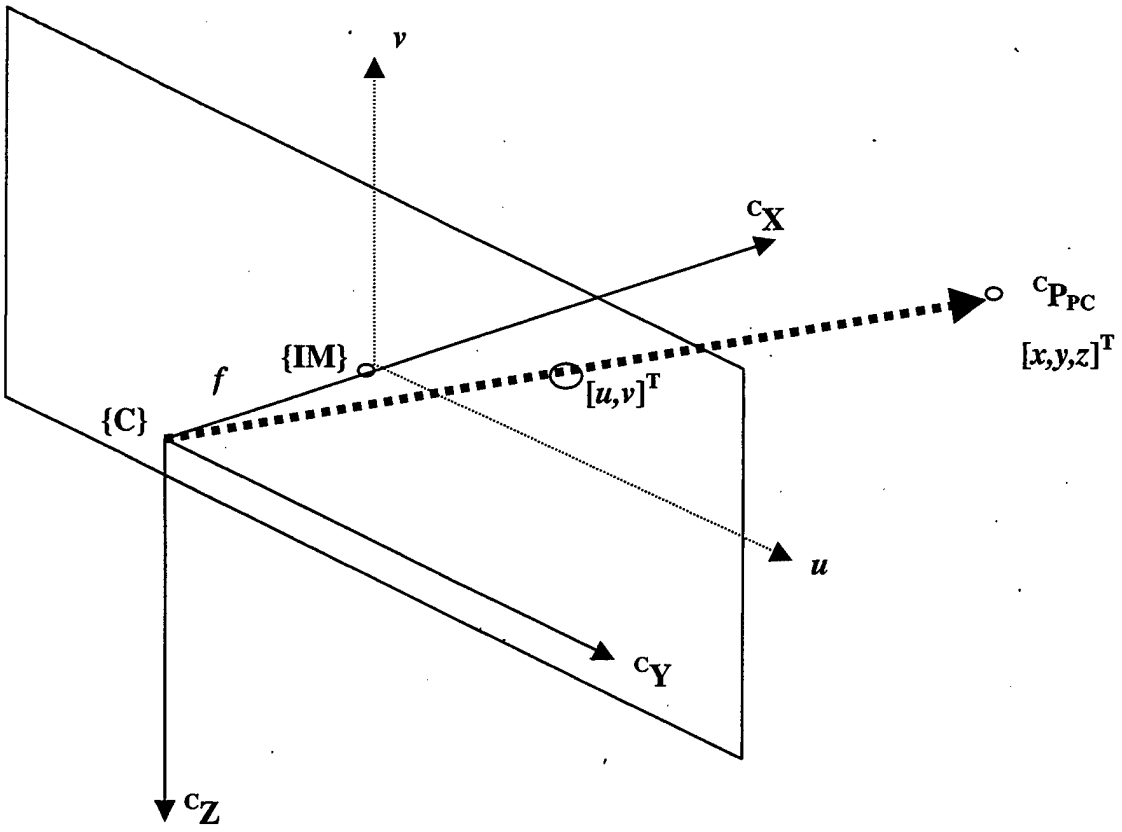

Figure 16 Image Plane Reference Frame

Let f be the focal length of the camera, $[u,v]^T$ denote the projection of ${}^C\mathbf{P}_{PC}$ onto the Image plane, and suppose ${}^C\mathbf{P}_{PC} = [x,y,z]^T$. Then

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{x} \begin{bmatrix} y \\ z \end{bmatrix}. \qquad \text{(III.8)}$$

IV. SENSOR MODEL

The development of the camera model was pivotal to the design of the vision controller. Not only was it required for the design phase, it was used in all the simulation and flight testing done during this project. The coordinate transformations from Chapter III were used extensively. The purpose of the model was to simulate the output from a camera mounted in the nose of the aircraft while it was tracking a curve on the ground. The ultimate goal was to simulate an "S" turn in the river next to the flight test airfield.

A. TRANSFORMATIONS

The sensor model transformations take a point along a curve given in LTP coordinates and convert it into the two dimensional output of the camera model.

Let

$\mathbf{P}_B$ = the position of the origin of {B} resolved in {U} (position of the aircraft),

$\mathbf{P}_P$ = the position of the point on the curve resolved in {U},

${}^B\mathbf{P}_C$ = the position of the camera resolved in {B},

$\mathbf{P}_C$ = the position of the camera resolved in {U},

$\mathbf{P}_{PC}$ = the relative position of the point with respect to the camera resolved in {U},

${}^C\mathbf{P}_{PC}$ = the relative position of the point with respect to the camera resolved in {C},

R = the various rotation matrices (Eqs. III.6, III.7),

$\pi({}^C\mathbf{P}_{PC})$ = the projection mapping of ${}^C\mathbf{P}_{PC}$ resolved in {IM} (Eq. III.8).

Figure 17 shows the relationship of the vectors with respect to the origin of both the LTP {U} and the Body reference frames {B}.

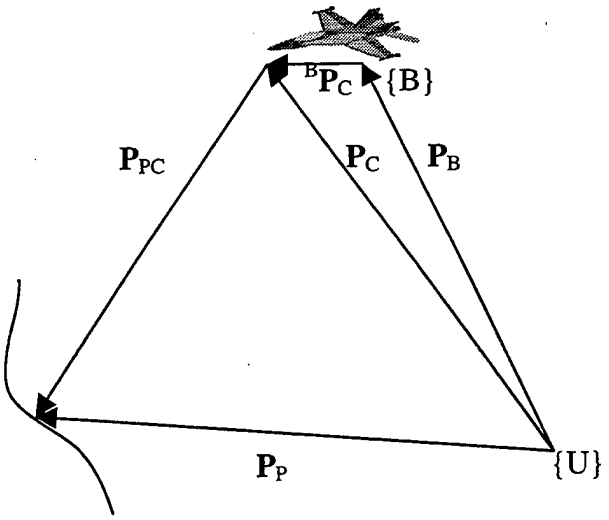

Figure 17 Position Vectors

From vector algebra it can be seen that

$$\mathbf{P}_C = \mathbf{P}_B + {}^{U}_{B}R\ {}^{B}\mathbf{P}_C \qquad \text{(IV.1)}$$

and

$$\mathbf{P}_{PC} = \mathbf{P}_P - \mathbf{P}_C \qquad \text{(IV.2)}$$

The coordinates of the point $\mathbf{P}_{PC}$ are then converted from inertial to Camera references frames by

$${}^{C}\mathbf{P}_{PC} = {}^{C}_{B}R\ {}^{B}_{U}R\ \mathbf{P}_{PC} \qquad \text{(IV.3)}$$

These are the three dimensional coordinates of the point as seen by the camera. The final transformation takes this result and converts it to the two dimensional Image plane

coordinates that are used by the controller. Using the pinhole camera model these coordinates are given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \pi({}^{C}\mathbf{P}_{PC}). \tag{IV.4}$$

Figure 18 shows the SystemBuild block diagram of the implementation of these equations. The noise source was added for test simulations.

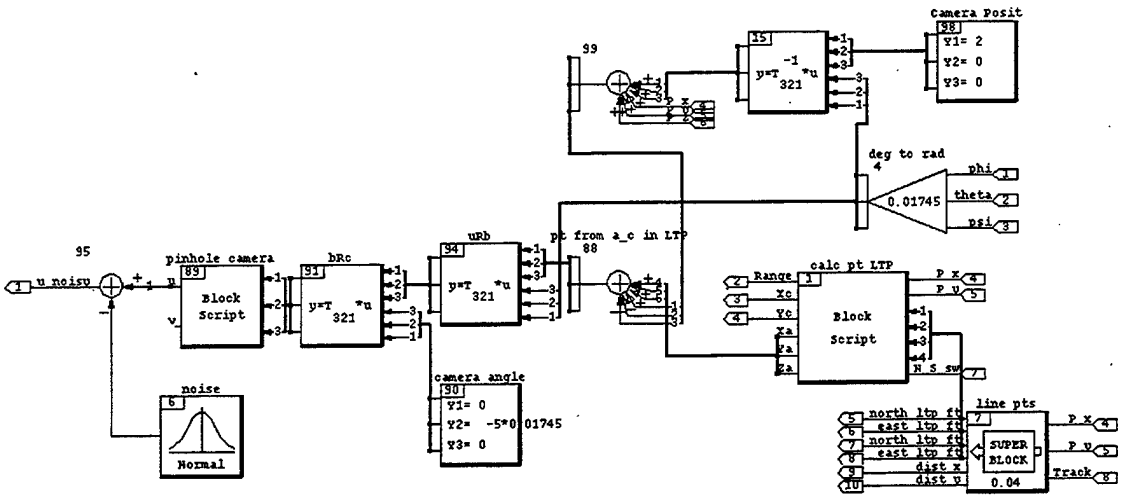

Figure 18 Camera Model

B. LTP CURVE POINTS

The process for calculating the LTP coordinates of the point on the curve varied depending on the type of curve being simulated. The initial implementation used a straight line as the desired ground track. Points along a circle and a sine wave were also calculated to simulate a non-linear track. The final simulation used a series of straight lines to simulate the course of the river shown in Figure 19.

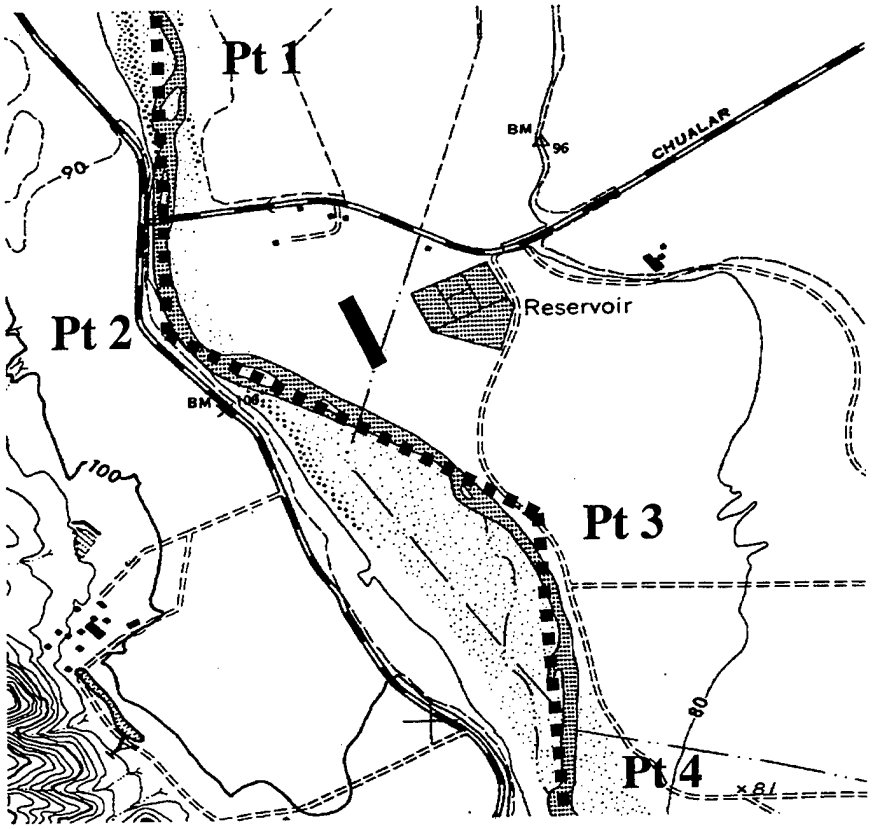

Figure 19 Flight Test Area

Each case used the aircraft's LTP position to calculate the LTP coordinates of the closest point on the curve. When the closest point is known, a point with a visibility distance of 1000 feet along the curve ahead of the aircraft is calculated. This is the LTP control point that is transformed into the Image plane and used by the controller. To ensure that the control point is in the simulated camera's field of view, the visibility distance must be calculated based on the field of view of the camera, the camera depression angle, and the altitude of the aircraft. This distance also increases the controller's stability, which will be discussed in Chapter V. Figure 18 shows the SystemBuild block diagram used when simulating a straight line. The user defined block titled "calc pt LTP" performs the control

point calculations. It takes as inputs the aircraft's position, the two points that define the line, and the direction of flight. The outputs are the LTP coordinates of the control point, the range to the line, and the coordinates of the closest point on the line. The last two parameters are used only to evaluate performance.

Let

$[X_F, Y_F]^T$ = the horizontal plane LTP coordinates of the FROG,

$[X_C, Y_C]^T$ = the horizontal plane LTP coordinates of the closest point on the line,

$[X_A, Y_A]^T$ = the horizontal plane LTP coordinates of the control point,

$[X_1, Y_1]^T$ = the horizontal plane LTP coordinates of point one,

$[X_2, Y_2]^T$ = the horizontal plane LTP coordinates of point two,

d = the visibility distance,

and

m = the slope of the line.

The closest point on the line to the FROG is found by the intersection of the line defined by the two points and the line perpendicular to it that passes through the FROG's position. Using the point slope equation for a line, the intersection coordinates are given by

$$Y_C = (Y_F*m\text{^}2 + X_F*m - X_2*m + Y_2)/(1 + m\text{^}2), \qquad \text{(IV.5)}$$

and

$$X_C = (Y_C - Y_2)/m + X_2. \qquad \text{(IV.6)}$$

The control point coordinates are given by

$$X_A = X_C + d*\cos(\tan^{-1}(m)), \qquad \text{(IV.7)}$$

and

$$Y_A = m*(X_A - X_C) + Y_C. \qquad \text{(IV.8)}$$

Similar user defined blocks are used to simulate a circle and a sine wave. The Xmath code used for the line, circle, and sine wave calculations are found in the Appendix.

C. RIVER SIMULATION

For the case of tracking of a single line, these points were fed directly into the "calc pt LTP" block. To simulate the river, however, required additional reference frame conversions and switching logic to approximate the desired "S" turn by the airfield. Figure 20 shows the SystemBuild blocks that perform these functions.

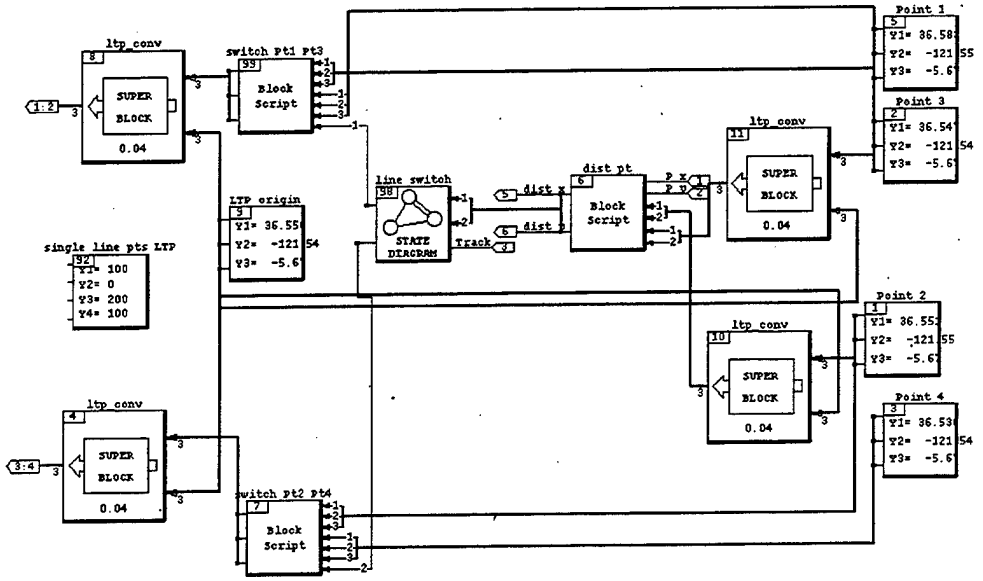

Figure 20 Line Points

The four points that define the three lines shown in Figure 19 were entered by their latitude and longitude as well as the position of the origin of the LTP. The points were converted to LTP coordinates in the blocks titled "ltp_conv" using Eqs. III.1 through III.4.

At system initiation points one and two were used to define the line. When the specified distance to point two was reached, points two and three were used to define the line. When the specified distance to point three was reached, points three and four were used to define the line. For ease of calculation, the distance to each of these switching points was done using LTP coordinates in the user defined block titled "dist pt". The logic to switch between these points was made through the use of a state machine titled "line switch" and two user defined blocks titled "switch Pt1 Pt3" and "switch Pt2 Pt4". When the conditions specified in the state machine to switch between points are met, it sends the signal to the user defined blocks to switch from point one to point three or from point two to point four.

D. STATE MACHINE

A state diagram block performs logical operations on the inputs defined by the user. These logical operations determine the transition criteria between the different states of the state machine. Transitions between the states only occur in the direction specified by the user. The state machine will not revert to a previous state unless the user specifically sets up a transition in that direction with its own transition conditions. In other words once the conditions are met to transition from one state to another, the state machine will remain at the new state even if the transition condition becomes invalid. In this application this feature allowed the use of decreasing aircraft range to the switching points as the transition criteria between the states. The state machine does not revert to tracking the previous line even after the aircraft passes the switching point and the range increases to where it no longer meets the original transition condition.

The state diagram block can have any number of outputs. All the outputs are Boolean in nature with a value of one for True and zero for False. In this application the outputs are used as flags in the user defined Xmath code blocks to switch between the four points used to define the three lines. The outputs can be set to True or reset to False any number of times. Also, the outputs are of two different types depending on where they are set or reset. An output that is changed upon reaching a new state is called a Moore output. An output that is changed during the transition between states is called a Mealy output. The Mealy output can be used when a state has more than one path leading to it, ensuring that the same conditions are set by the machine at the arrival at a given state no matter what path is followed getting there.

The diagram shown in Figure 21 is the state machine used to switch between the three lines. At system initiation all outputs are set to False. With both outputs False the user defined switching blocks output points one and two, which define line one. When the vision controller switch is on and the FROG is less then 700 feet to the first line, the machine transitions to the first state labeled "Track line 1". No outputs are changed at this state. When the FROG is less than 600 feet to point two, the machine transitions to state two labeled "Track line 2". At this state a Moore output is used to set output one to True. With output one set to True the switching blocks will output points two and three, which define line two. When the FROG is less than 600 feet to point three, the machine transitions to state three labeled "Track line 3". At this state both outputs one and two are set to True. With both outputs set to True the switching blocks will output points three and four, which define line three.

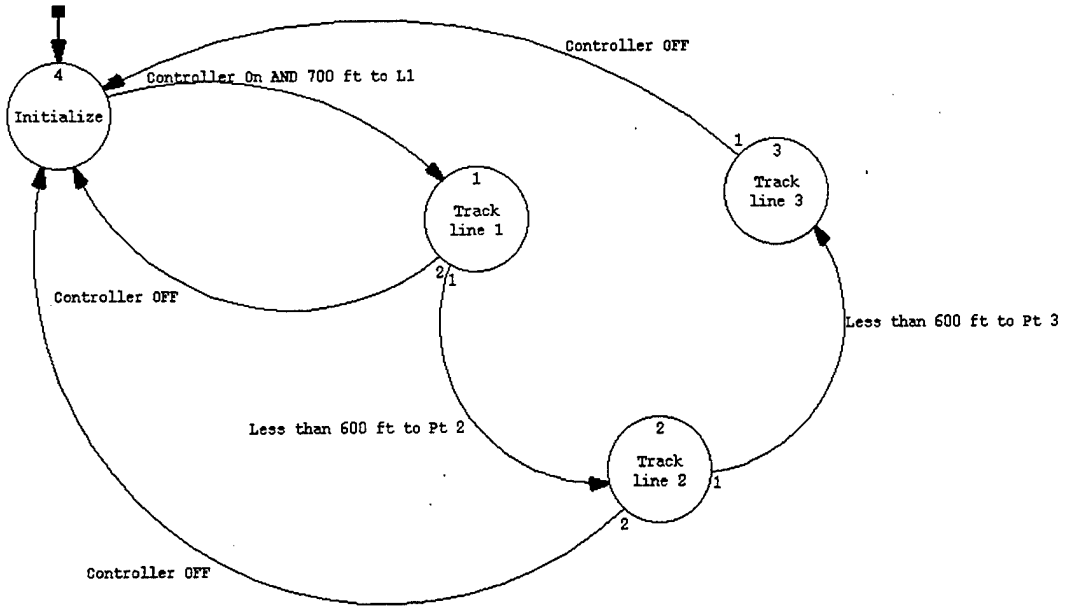

Figure 21 State Machine

If the vision controller switch is turned off at any time, the machine transitions back to the initialization state. When this happens from state one, no outputs are reset since none were set at state one. When this occurs from state two, a Mealy output is used to reset output one to False, since this was the only output set upon arrival at state two. When this occurs from state three, Mealy outputs are used to reset both outputs one and two to False. This demonstrates the use of Mealy versus Moore outputs. The transition back to the initialization state means that the points for line one are again used to define the desired track. Thus, the resetting of the vision controller is accomplished without having to turn off the entire RealSim controller operation, which is an important operational feature.

V. VISION CONTROLLER DEVELOPMENT

A. DESIGN CONCEPT

Let $[u,v]^T$ denote the Image plane coordinates of the control point along the line representing the river. Suppose $u = 0$, then in straight and level flight the x-z plane of the both the aircraft's Body and Camera coordinate systems are coincident with the vertical plane that contains both the point along the river and the aircraft. As shown in Figure 22, this places the control point directly in front of the aircraft. Additionally for the case of a straight line, if $u = 0$ and $du/dt = 0$, then the aircraft is on the line.

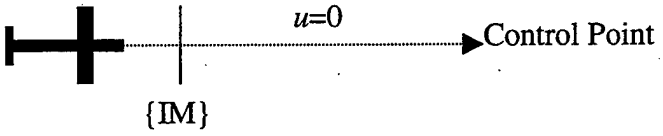

Figure 22 Geometry of $u = 0$

Therefore, the control strategy was to drive u to zero using yaw rate command as shown in Figure 23.

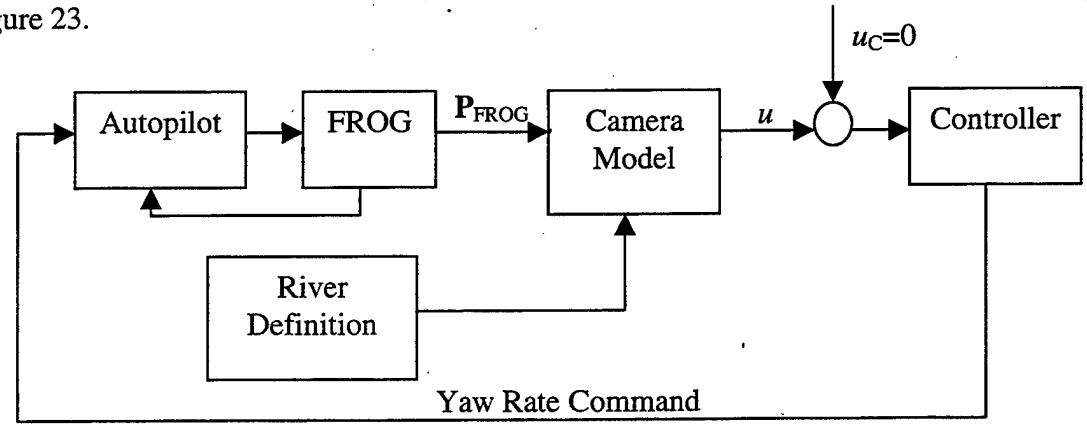

Figure 23 Control Strategy

With this design concept in mind, the vision controller design requirements are:

1. Seamless Transition – no large fluctuations when engaged.
2. Automatic Capture and Track of the desired course.
3. Integration into the FMS that allows use of existing controllers.
4. Stable Feedback system.
5. Transient performance rapid enough to be evaluated in the confines of the flight test area.
6. Steady State errors small enough to keep the desired track in the field of view of the camera.
7. 6 dB positive Gain and 45° Phase stability margins in the control loop.

B. DESIGN METHOD

The control system design approach used was the root-locus method described in Ogata [Ref. 5] aided by the computational capability of Xmath. This method uses a graphical approach for finding the position of the closed-loop poles based on knowledge of the locations of the open-loop poles and zeros. If the desired performance can not be achieved by merely adjusting the gain in the feedback loop, adding a compensator in the feedback system reshapes the root loci. The compensator adds to the system additional poles and/or zeros that influence the shape of the root loci so as to meet the desired performance. This is a trial and error approach to system design that is greatly enhanced by Xmath, which can quickly recalculate and display the root-locus plot for changes to the system.

C. CONTROLLER DESIGN

1. Initial Design

The model for the FROG/Autopilot plant combination was developed in Papageogiou [Ref. 6] and has been used in many prior theses and class projects. The sensor model and FROG/Autopilot plant combination is shown in Figure 24.

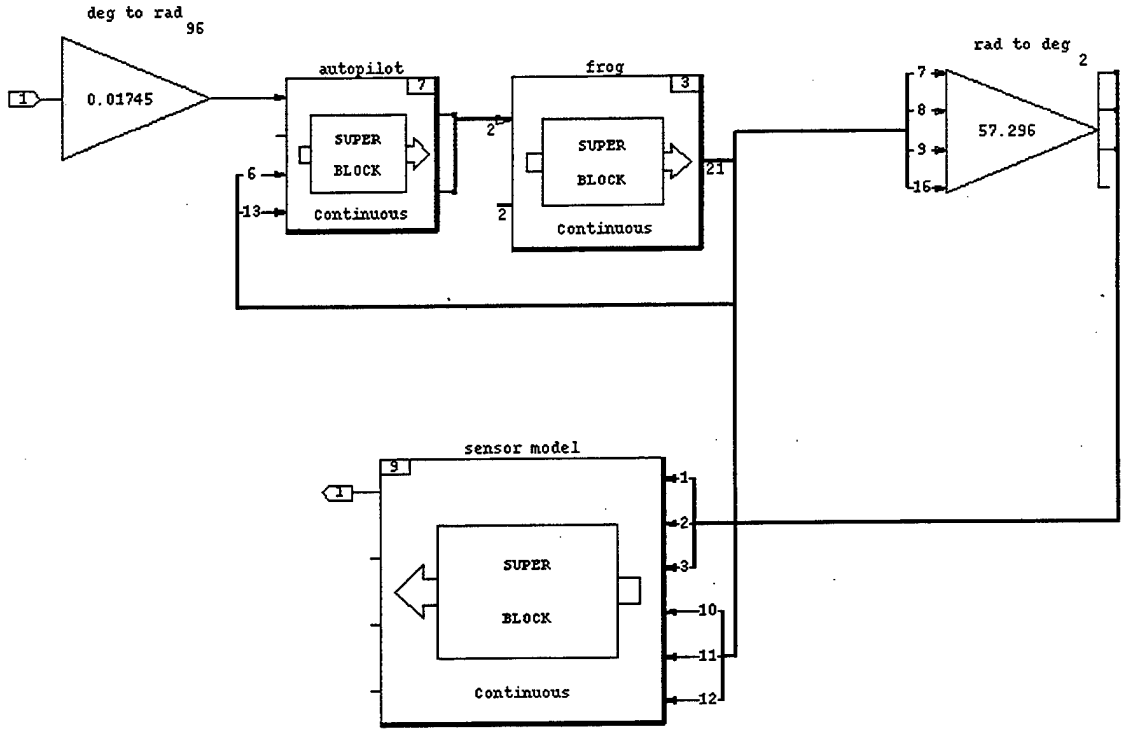

Figure 24 FROG/Autopilot/Sensor Plant

The sensor model for the initial design used a single straight line for the desired track. Using this model the system was linearized around a trim point with the FROG in straight and level flight with a velocity of approximately 50 knots. The open-loop eigenvalues are shown in Table 1 and the root-locus plot in Figure 25. The eigenvalues of

concern are the complex pair with a damping of 0.41 and frequency of 0.72, which quickly migrate to the right half plane causing the system to go unstable.

Eigenvalue	Damping	Frequency (rad/sec)
0	-1.0	0
0	-1.0	0
0	-1.0	0
0	-1.0	0
-0.1687	1.0	0.1687
-0.2972±0.6594i	0.4109	0.7232
-2.9602	1.0	2.9602
-4.0115	1.0	4.0114
-1.6310±3.8123i	0.3933	4.1466
-0.8426±4.1593i	0.1985	4.2437

Table 1 OL Plant Eigenvalues

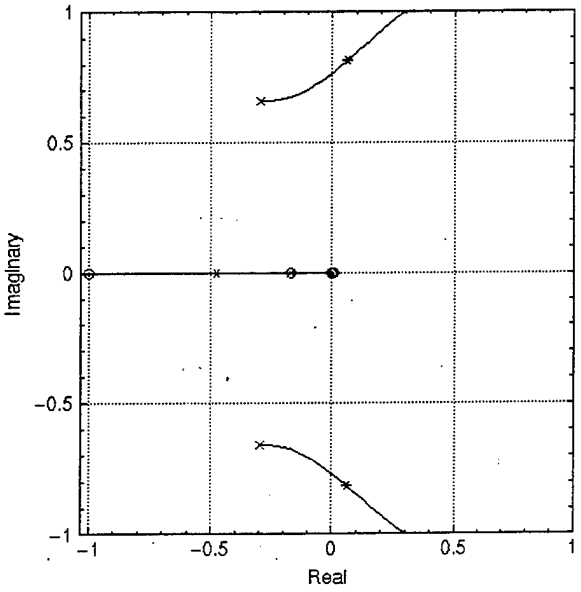

Figure 25 Root-locus OL Plant

The initial sensor model used the closest point on the curve from the aircraft as the control point. The rationale was to try to drive the FROG to the closest point on the line. This approach is not desirable for two reasons. The first is that when the FROG is properly tracking the line the control point is directly underneath the aircraft. Though this point

called the Nadir theoretically exists in {IM}, it would not be in the field of view of the camera. The second problem with this approach is that it produces a nonminimum phase zero very close to the origin. A zero at this location makes it hard to control the poles of concern. The final control point uses a visibility distance of 1000 feet along the curve in front of the FROG. This places it the camera field of view and moves the zero to .0656 as shown in Figure 26. Though this is still a nonminimum phase system, it allows the poles of concern to be controlled as desired.

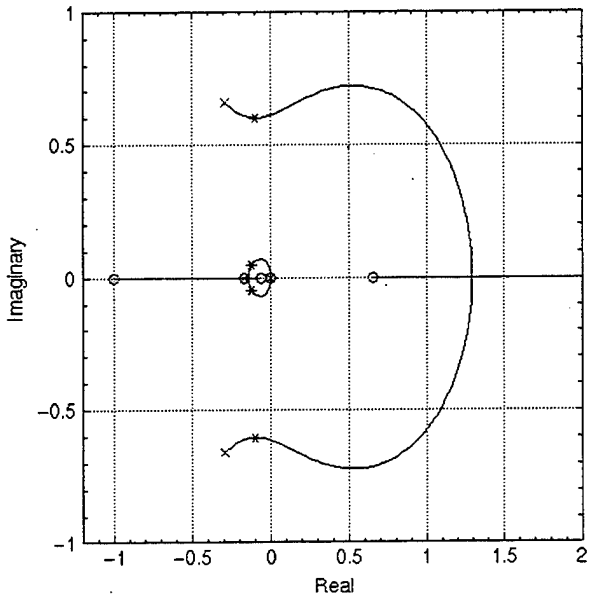

Figure 26 Root-locus New Sensor

With the visibility distance set, a single pole and zero compensator with the transfer function of

$$H(s) = \frac{-0.1s + 1}{s + 1} \qquad \text{(V.1)}$$

was added to the system. This adds a zero at positive 10 and a pole at –1.0 that cancels a zero from the plant at that location. The effect to the root-locus plot is shown in Figures 27 and 28. The eigenvalues of concern are drawn down towards the Real axis and remain stable for much greater values of the gain. The gain shown in Figure 28 is 75. The closed-loop eigenvalues for this system are stable and shown in Table 2. This simple feedback compensator design also has the same transfer function as the lateral component of the autopilot in the FROG/Autopilot plant.

Eigenvalue	Damping	Frequency (rad/sec)
0	1.0	0
0	1.0	0
-0.1199	1.0	0.1199
-0.1687	1.0	0.1687
-0.2515	1.0	0.2515
-0.1335±0.4514i	0.2837	0.4707
-1.0	1.0	1.0
-2.9602	1.0	2.9602
-3.9593	1.0	3.9593
-1.6310±3.8123i	0.3933	4.1466
-0.8466±4.1670i	0.1991	4.2521

Table 2 CL Compensated Eigenvalues

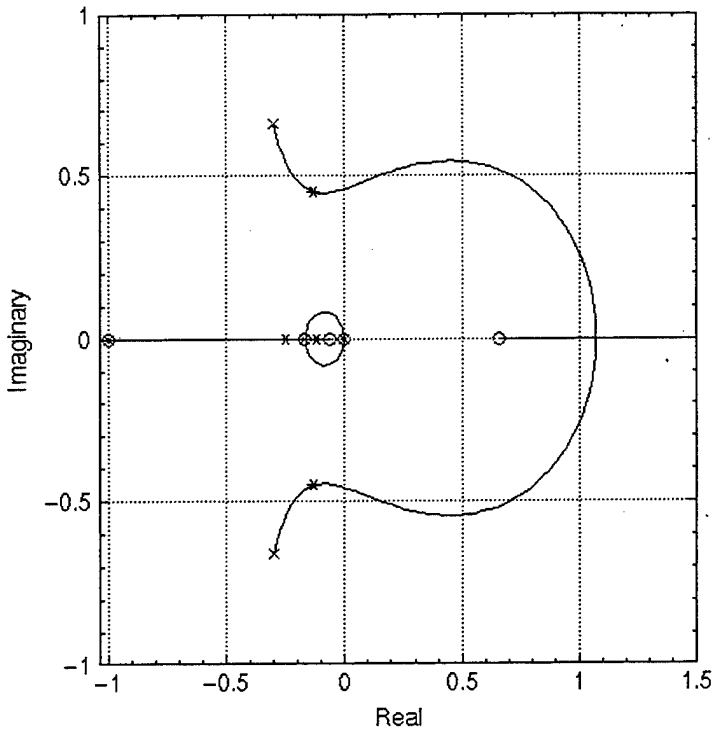

Figure 27 Root-locus OL Controller

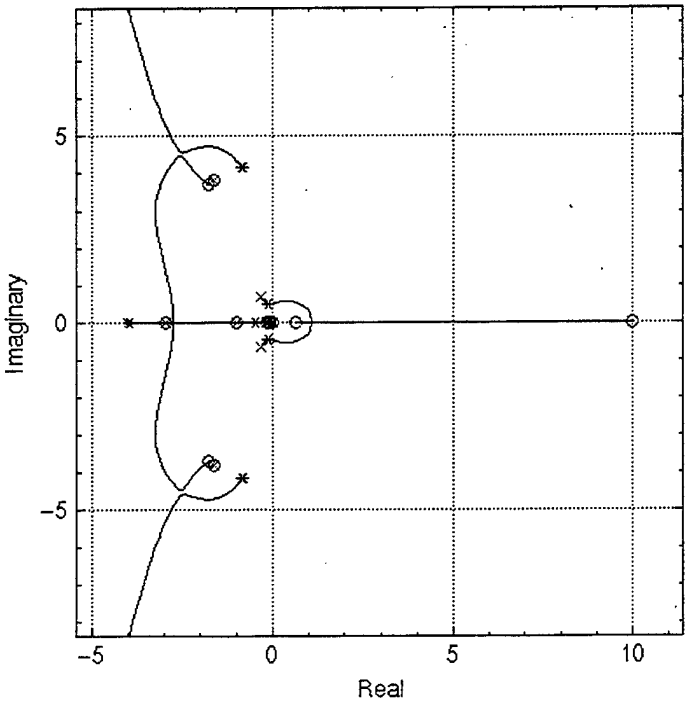

Figure 28 Root-locus OL Controller Enlarged

2. Frequency Analysis

With the initial control design complete, the system as shown in Figure 29 was linearized for robustness analysis. The Bode plot of this system is shown in Figure 30. The bandwidth of the system is seen to be 0.22 rads/sec. This is sufficiently below the yaw rate response bandwidth of 1.2 rads/sec discussed in Rivers [Ref. 3]. The stability margins with the feedback gain set at 75 are 3.68 dB of gain and 35° of phase margin. These are below the desired margins specified in the design goals, which are commonly used in controller design. Reducing the feedback gain to 55 achieves the specified stability margin design goals.

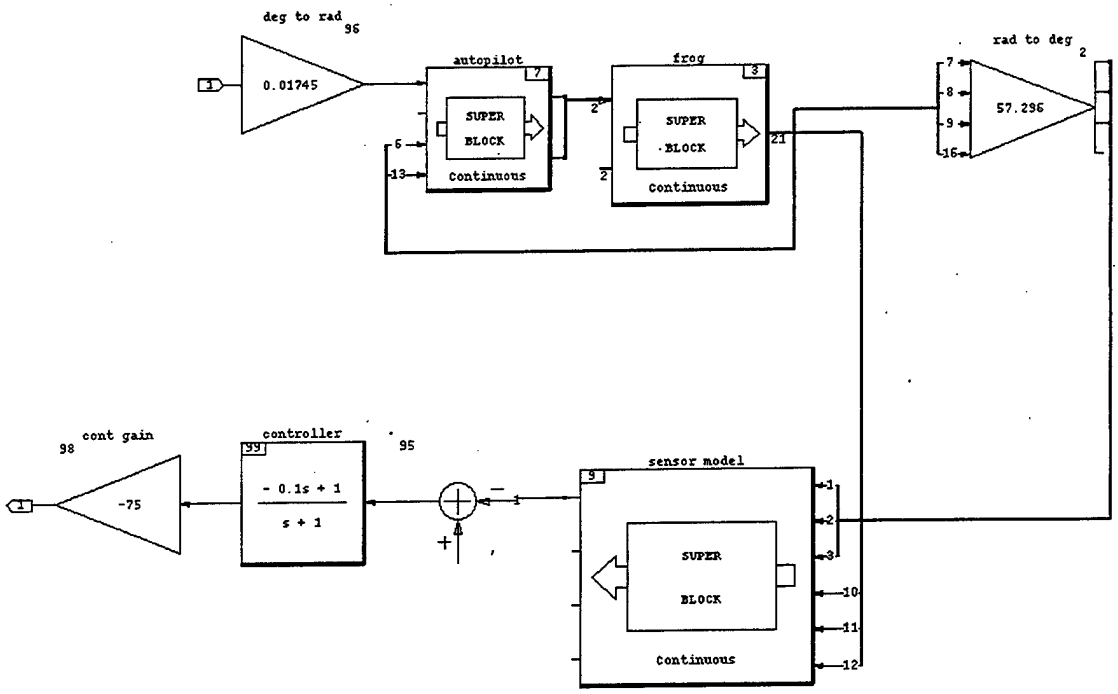

Figure 29 Control Loop System

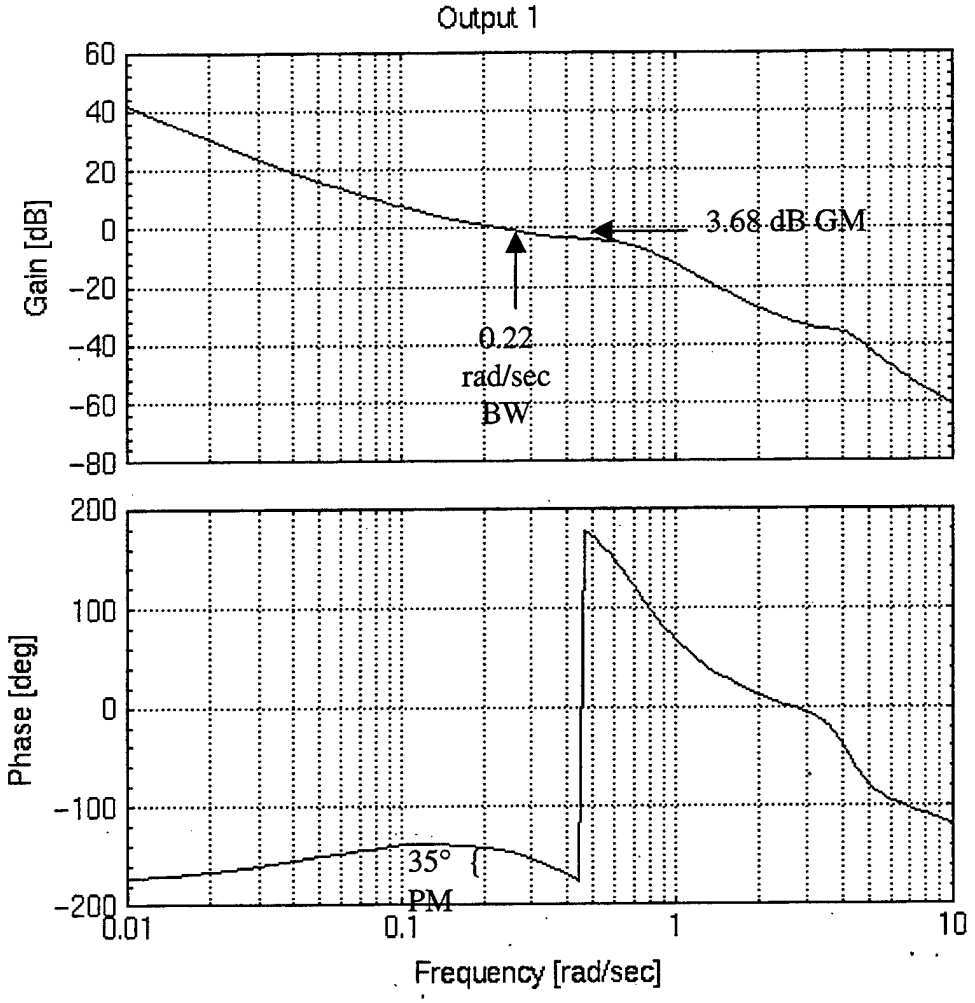

Figure 30 Control Loop Bode Plot

To determine the command loop bandwidth the feedback loop is closed, and the system is linearized as shown in Figure 31. Figure 32 shows the Bode plot of the command loop. The bandwidth of the system is seen to be 0.6 rads/sec, which is again below the yaw rate response bandwidth.

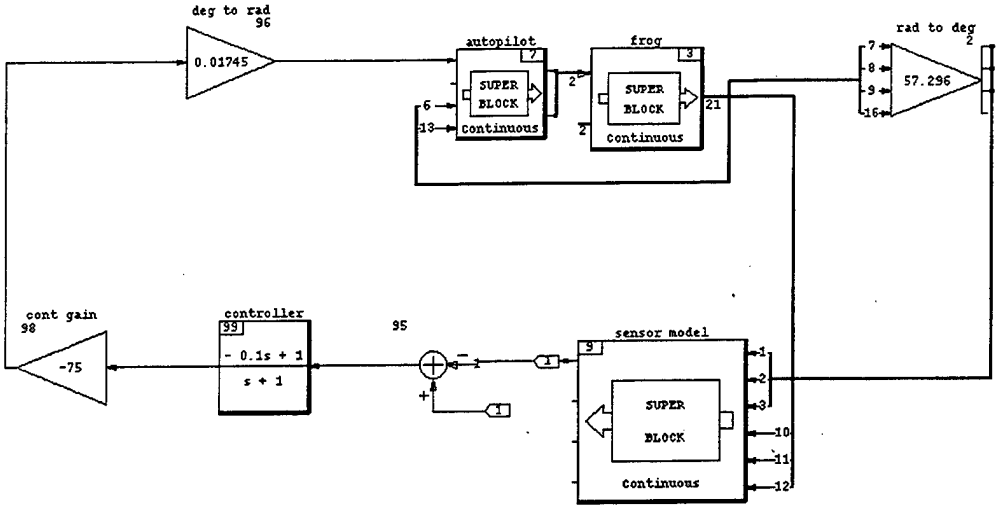

Figure 31 Command Loop System

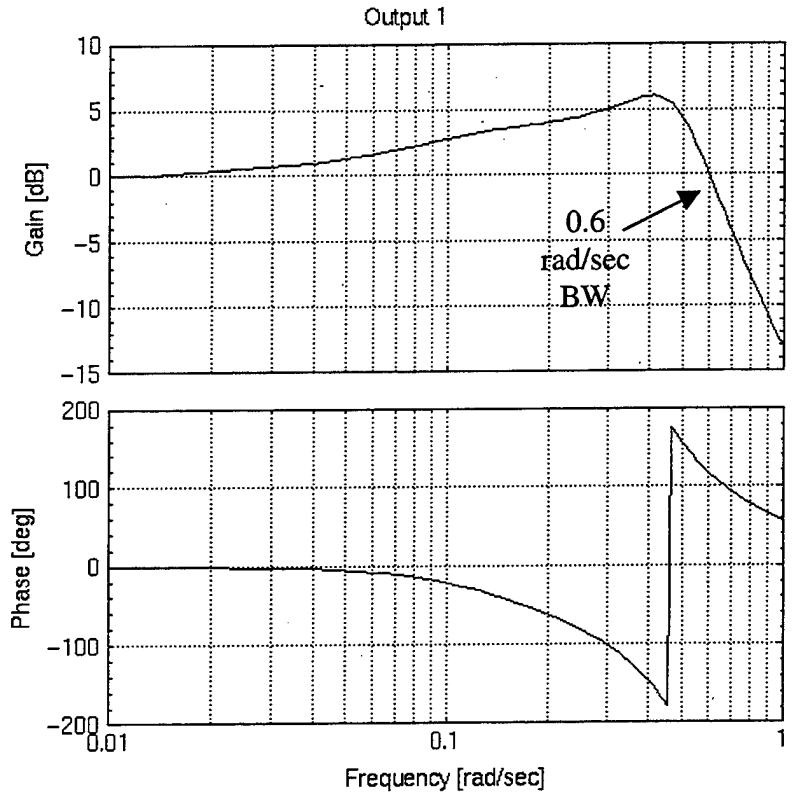

Figure 32 Command Loop Bode Plot

3. Controller Response

No exact rise time or percent overshoot were specified for the transient response characteristics; however due to the confined area used for flight testing, the response must be sufficiently quick to capture and track the desired course in a very short time. Consequently, variable gains were tested to achieve the desired response. Test results will be detailed in Chapter VI. To summarize the results: a gain of 55 provided the desired stability margins; however, the transient response was such that the FROG was not be able to capture the line before the turn point was reached. A gain of 75 provided quick enough response, and it has adequate stability margins shown earlier. This gain was used in the initial flight tests.

4. Orientation

The feedback system consisting of the FROG, Autopilot, camera model, and vision controller is highly non-linear. The stability of this system depends on the chosen trim conditions. The position and orientation of the aircraft with respect to the line were found to greatly influence the stability. Obviously, a vision controller must keep the desired track in its field of view in order to work. In addition to this constraint, it was found that there is a cross track angle that will cause the system to go unstable. Cross track angles of 45°, 50°, 55°, and 60° were tested to determine the stability region. Table 3 shows the eigenvalues for the 55° case. As shown, the system with this cross track angle has an unstable complex pair. The implementation of the controller must take this into account. It was decided to

impose an implementation constraint of flying the aircraft into a 45° cone of the desired track before engaging the vision controller.

Eigenvalue	Damping	Frequency (rad/sec)
0	-1.0	0
0	-1.0	0
-0.0482	1.0	0.0482
-0.1687	1.0	0.1687
0.0272±0.4934i	-0.0550	0.4941
-0.6674	1.0	0.6674
-1.0	1.0	1.0
-2.9602	1.0	2.9602
-3.9308	1.0	3.9308
-1.6310±3.8123i	0.3933	4.1466
-0.8495±4.1740i	0.1994	4.2596

Table 3 55° Cross Track Angle

5. Discrete Transformation

Until now all design, testing, and analysis has been done with a continuous time system. For use in flight the controller must be discretized. SytemBuild performs most of this automatically. Time delays were manually added to all integrators to prevent algebraic loops. Additionally, differentiators were approximated or transfer functions manipulated to eliminate the zeros. Figure 33 shows the discrete transformation of the controller after its dynamics have been implemented as follows:

$$H(s) = \frac{-0.1s+1}{s+1} = -0.1\left(1 - \frac{11}{s+1}\right). \qquad \text{(V.2)}$$

After transformation the vision controller and the camera model were combined into a superblock for testing with the discrete model of the FROG/Autopilot plant in conjunction with the existing heading controller. Figure 34 shows the SystemBuild block diagram of the complete system used for the discrete simulations.

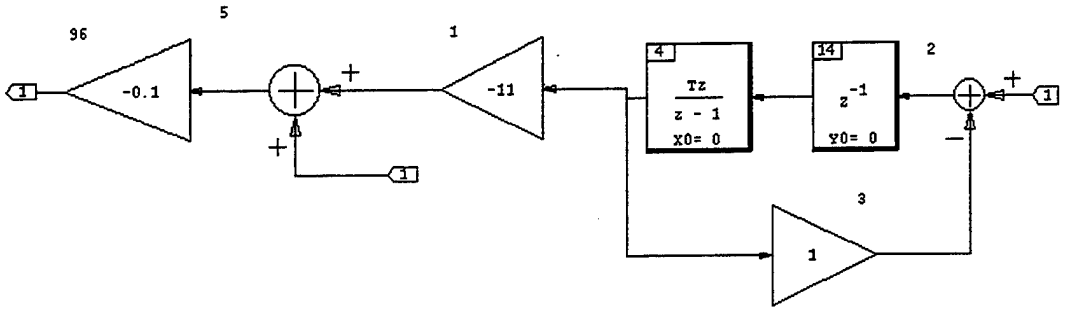

Figure 33 Discrete Controller

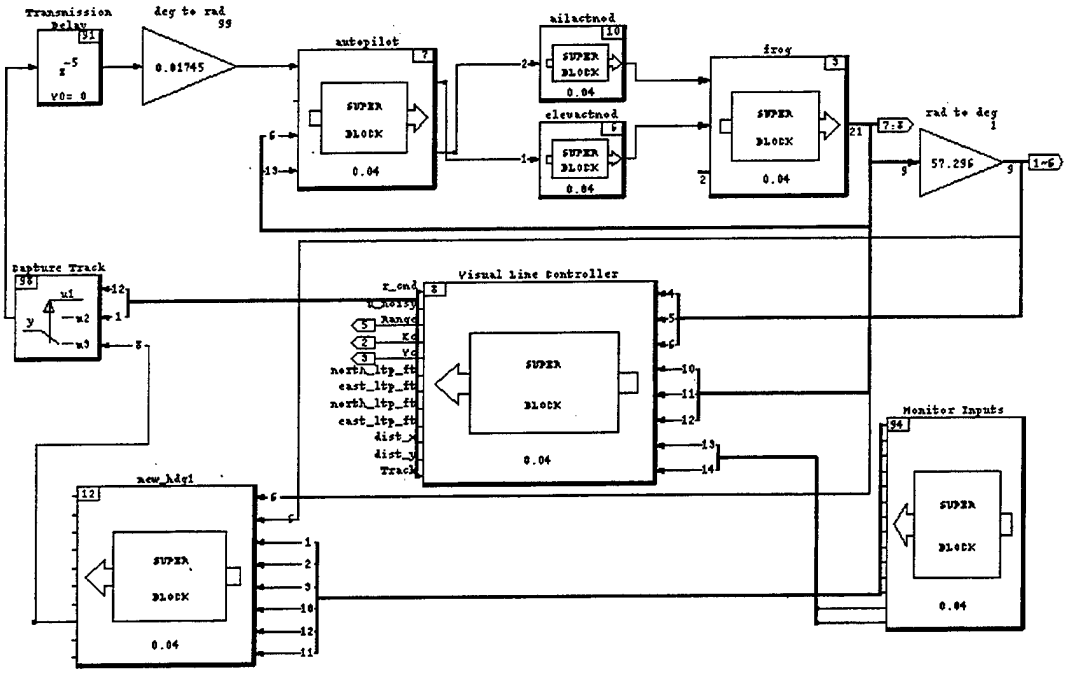

Figure 34 Complete Discrete Model

D. CONTROLLER INTEGRATION

A second state machine was designed to provide an automatic capture and track capability. This machine controls the "Capture Track" switch shown in Figure 34, which switches between the heading and vision controllers when the specified conditions are met. Figure 35 shows the state diagram block inside the "Visual Line Controller" super block.

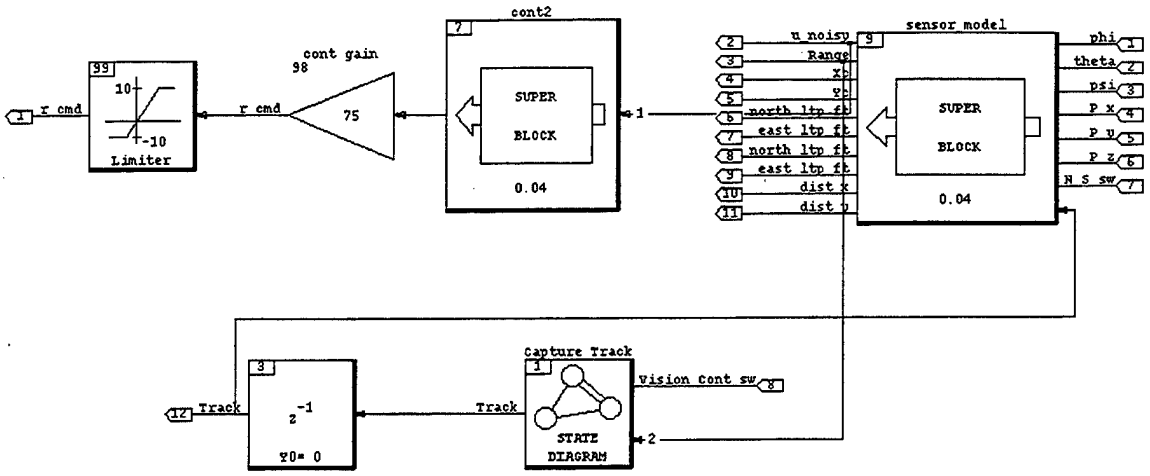

Figure 35 Visual Controller Super Block

This state machine was not incorporated into the state machine in the sensor model since it will be used when the actual onboard camera is operational and the sensor model is discarded. Figure 36 shows the diagram of the state machine.

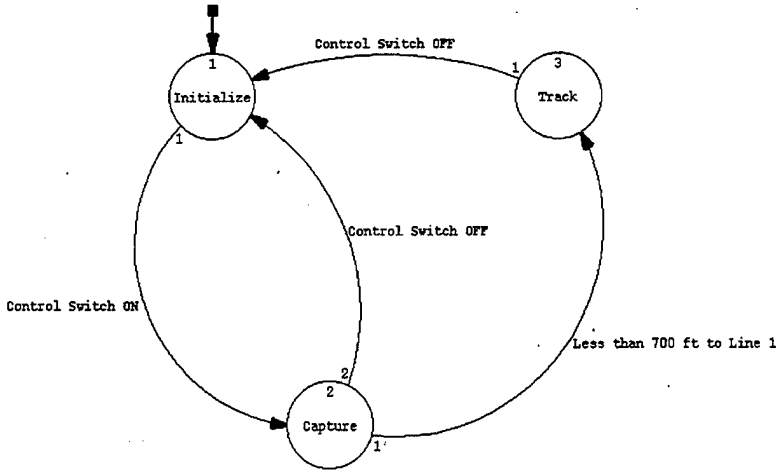

Figure 36 Capture/Track State Machine

At system initiation state one is active and the single output of the state machine is initialized to False. When the vision control switch is on, the machine transitions to state two. The output is not changed. When the range to the first line is less than 700 feet, the machine transitions to state three. At state three the single output is set to True by a Moore output. This output switches the yaw rate commanded source from the heading controller to the vision controller. If at any time the vision control switch is turned off, the machine transitions to the initialization state, and the output is reset to False causing the yaw rate commanded to revert to the heading controller.

Figure 37 shows the FMS animation page used for flight testing of the vision controller. The operator has the same controls for heading and altitude commands, including the Absolute/Delta options, as detailed in Chapter II. The throttle controller is not used for these tests. Digital readouts of all the parameters are displayed. Additionally, a graphic display of the FROG's position relative to the airfield and river are shown. The aircraft's initial position and movement scaling are set via the three sliders titled "Xo", "Yo", and "scale". The vision controller is engaged with the "Vision Control" switch, and the direction of flight is entered by the "North/South" pushbutton. The direction of flight is needed for the sensor model and will be eliminated when the actual camera is used. The final display is a bar indicator that shows when the vision controller is operating in the capture state, tracking line one, line two, or line three state (Cap/L1/L2/L3). An additional heading display is also shown that was used for some other off-line testing.

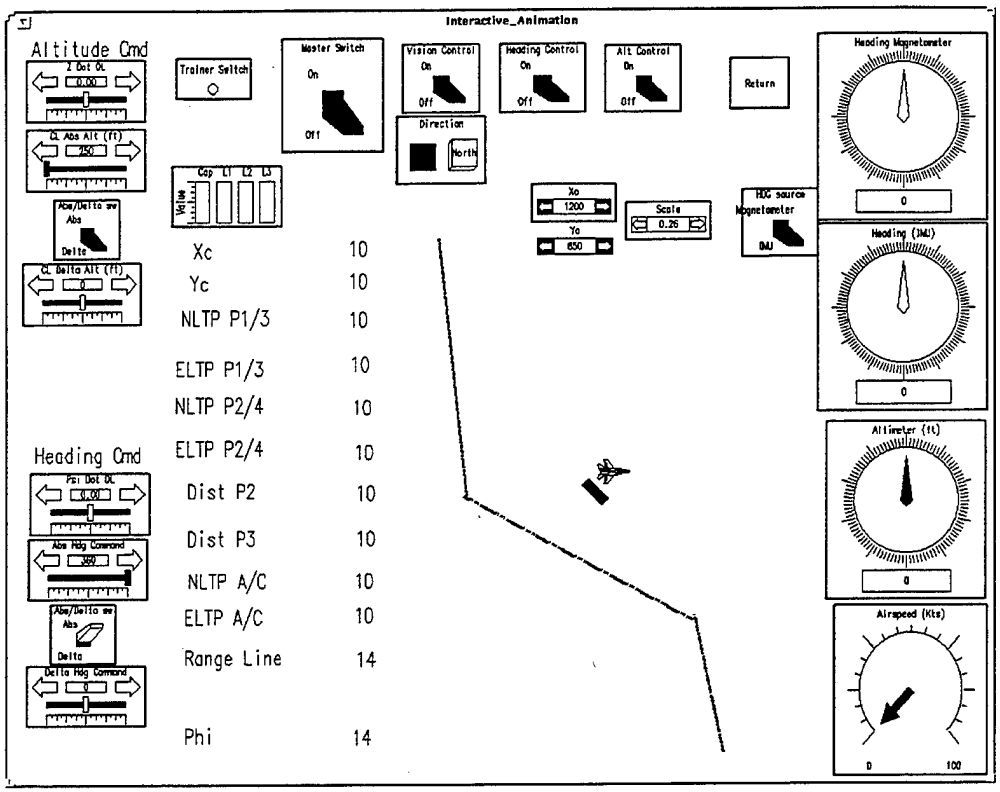

Figure 37 Vision Controller FMS Page

VI. TEST RESULTS

A. GROUND SIMULATION

1. Straight Lines

The first simulations used the single line sensor model that was used in the linearization during controller development. These tests not only validated the controller design, but they were also used for determining the optimum gain as discussed in Chapter V. Figure 38 shows the results for gains of 100, 75, and 50. All three gains are stable and show proper tracking of the line. Though a gain of 50 provides the desired stability margins, the transient response is so sluggish that the FROG would not be able to capture the line before the turn point is reached. A gain of 100 shows the fastest response; however, this would lower the stability margins below acceptable levels.

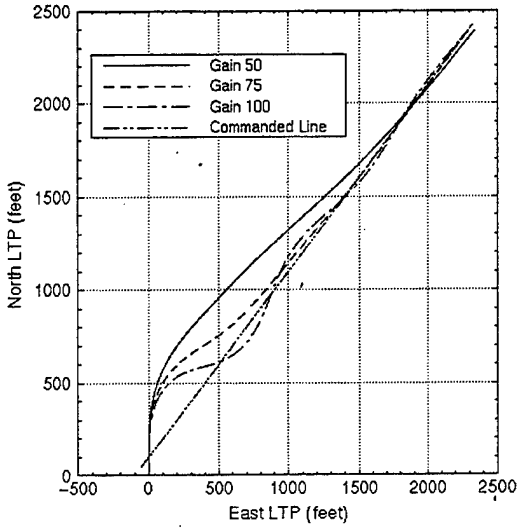

Figure 38 Line Tracking

Further gain tests were done using the simulated river model. Figure 39 shows the tracking error over the simulated river course for a gain of 55. Figure 40 shows the same for a gain of 75. The large spikes in the tracking error correspond to the state machine switching to a new line. A gain of 55 causes overshoots of 150 feet, and the aircraft does not have enough time to recapture the track during the first two legs. This performance is unacceptable for operational use. A gain of 75 shows overshoots of 15 feet, and the aircraft has plenty of time to recapture the track. A gain of 75 clearly provides a better response, and it has adequate stability margins listed in Chapter V. As stated previously, this gain was used for the initial flight testing.

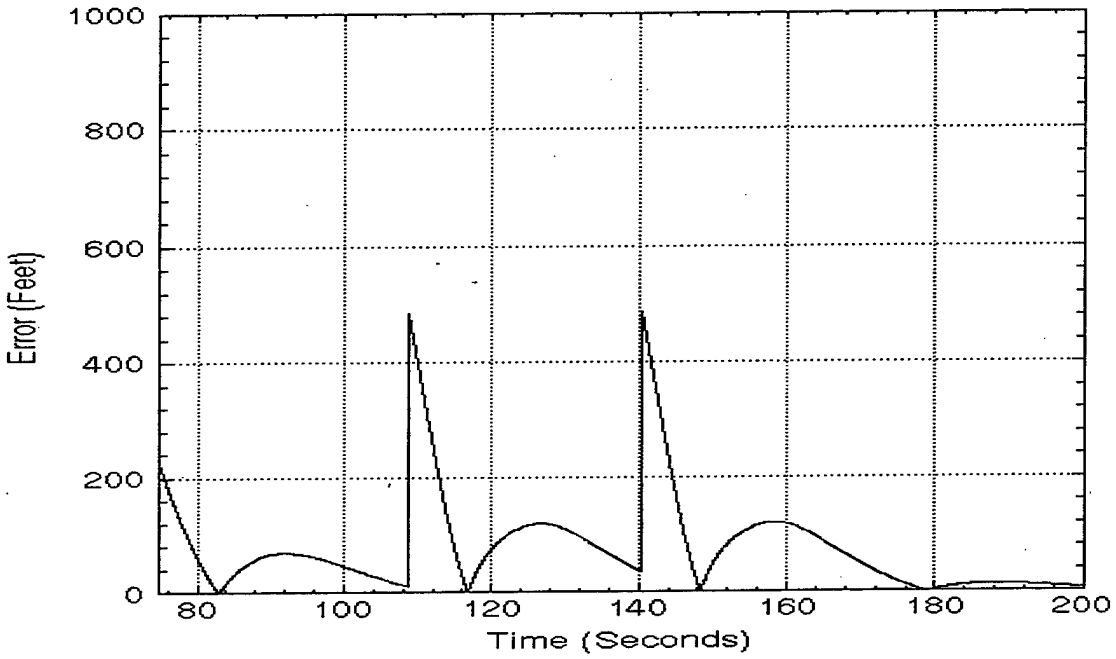

Figure 39 Tracking Error Gain 55

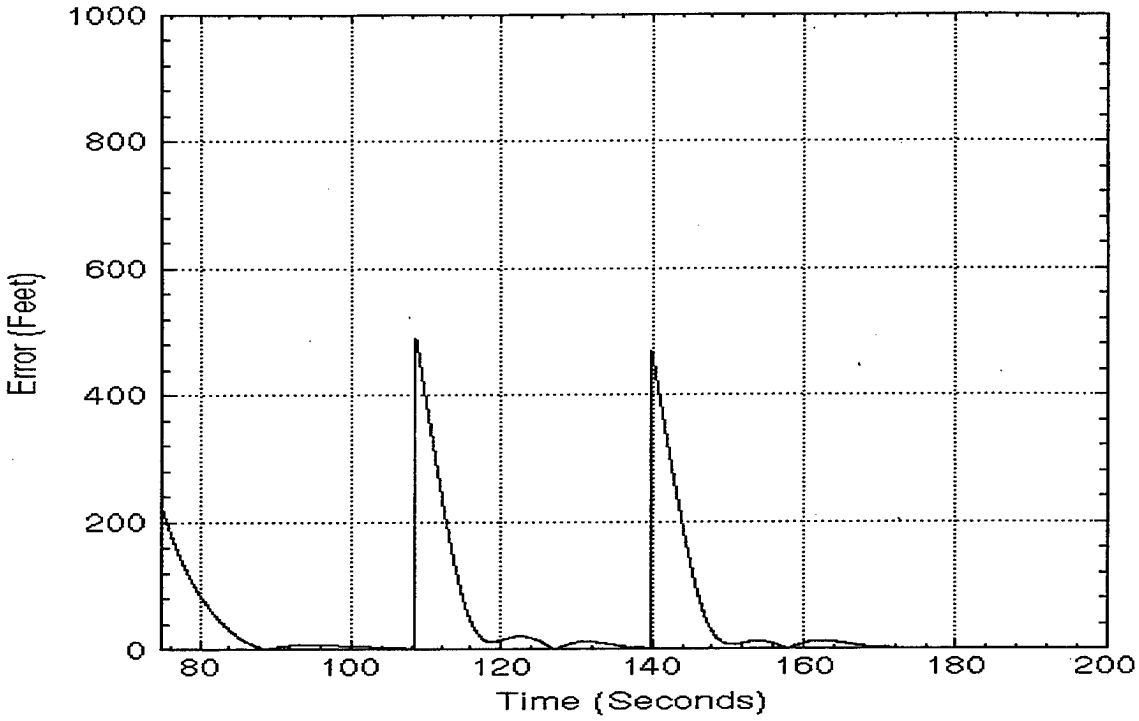

Figure 40 Tracking Error Gain 75

Figure 41 shows a plot of the flight path and the control point for a simulation of the fully integrated vision controller using the three line river model. Since the control point is on the desired track, it is plotted to represent the river. Jumps at the end of each line are a plotting anomaly that occurs when the control point switches to the new line.

The FROG was initialized 4500 feet north and 2500 feet east of the LTP origin on a heading of 330°. The heading and vision controllers were engaged. A heading of 210° was commanded via the heading controller to fly the FROG into a 45° cone of line one. At 700 feet from line one the vision controller switched states from "Capture" to "Track" and took over from the heading controller. Line one was tracked until 600 feet from point two, when the state machine switched points to compute line two in the sensor model. A smooth turn

was made with 15 feet of overshoot (Figure 40) to track line two. Line two was then tracked until within 600 feet of point three, when the sensor model switched to line three. Again a smooth turn was made with minimal overshoot to track line three.

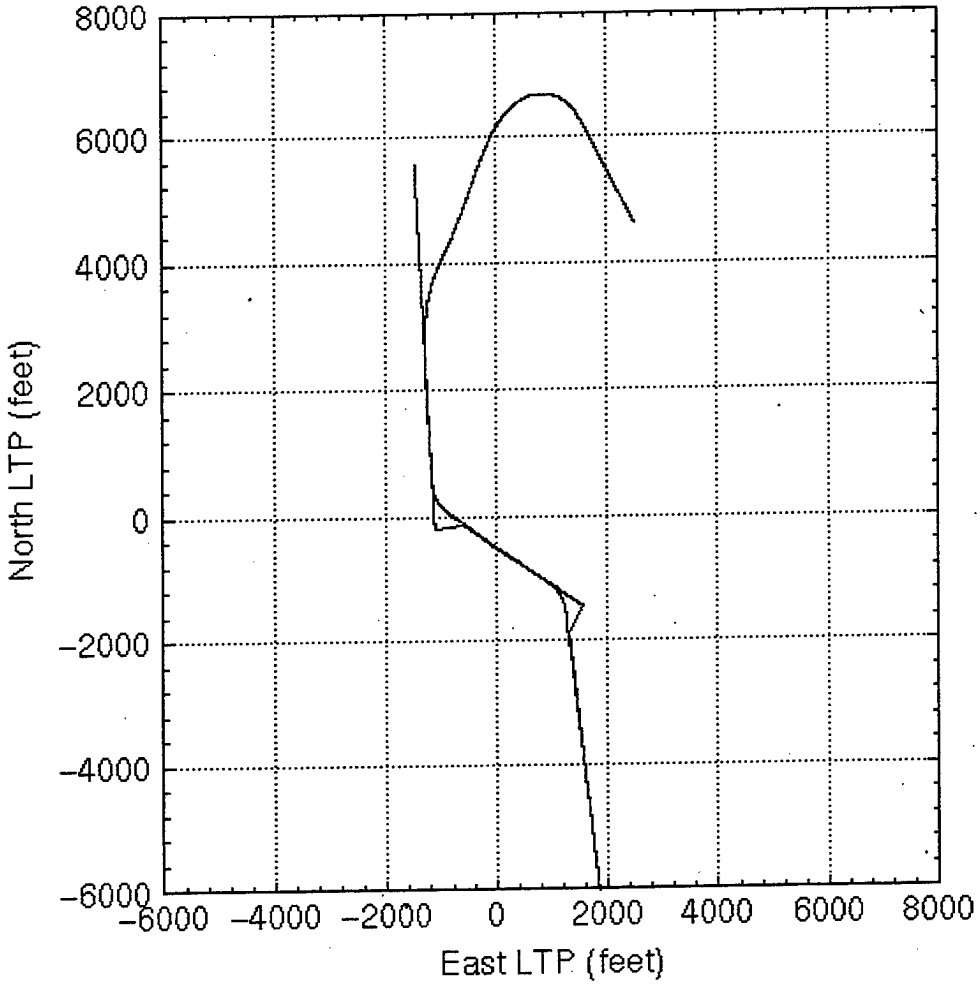

Figure 41 Full River Simulation

Figure 42 shows the sensor model and controller outputs for this simulation. The controller is exhibiting proper performance as evidenced by a positive commanded yaw rate

for a positive value of u from the pinhole camera model. Similarly, a negative yaw rate was commanded for a negative u value. Maximum commanded yaw rates occur at the switching points between the lines. Though no yaw rate over 10 deg/sec was commanded, a rate limiter of ±10 deg/sec was added as a safety factor for actual flight tests.

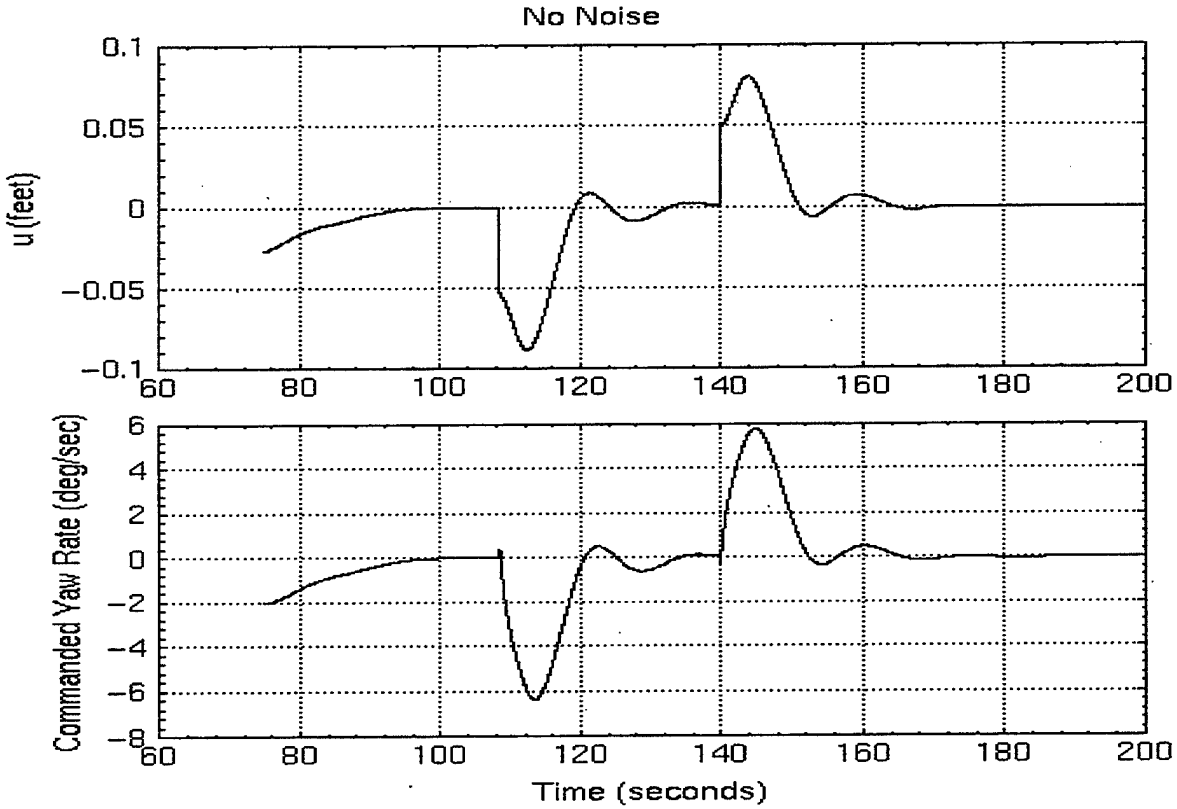

Figure 42 Camera/Controller Output

2. Simple Curves

The vision controller was designed to track a single line using a linearized plant model trimmed at a straight and level flight condition. A controller that remains stable only under these conditions would be of little operational value. The three line river simulation showed that the design is capable of tracking a more complex shape. To further check the

robustness of the controller design a commanded track for both a circle and a sine wave were tested. Figure 43 shows the flight path for a circle when the aircraft is initially outside the circle, and Figure 44 shows the flight path starting inside the circle.

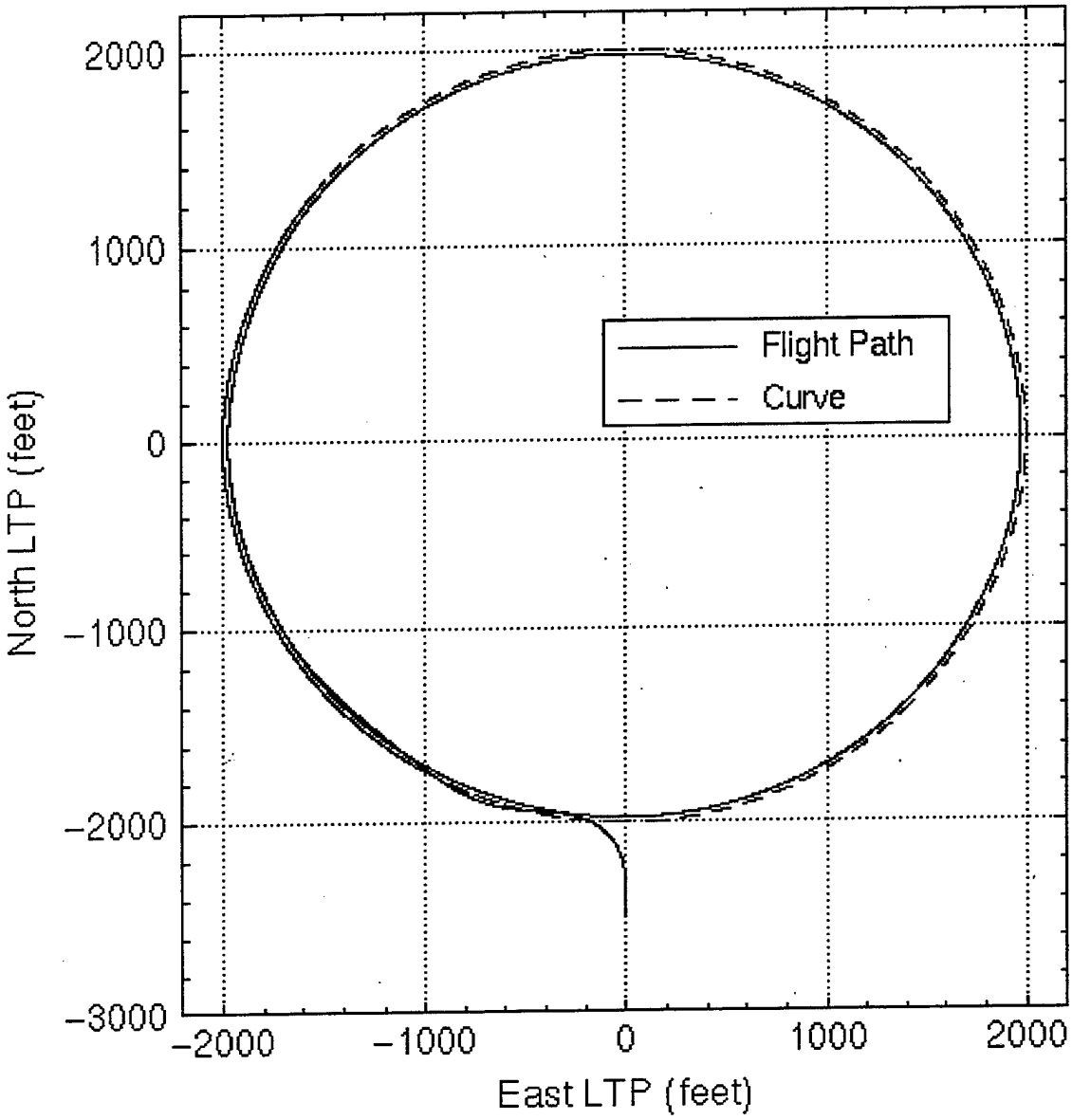

Figure 43 Outside Circle

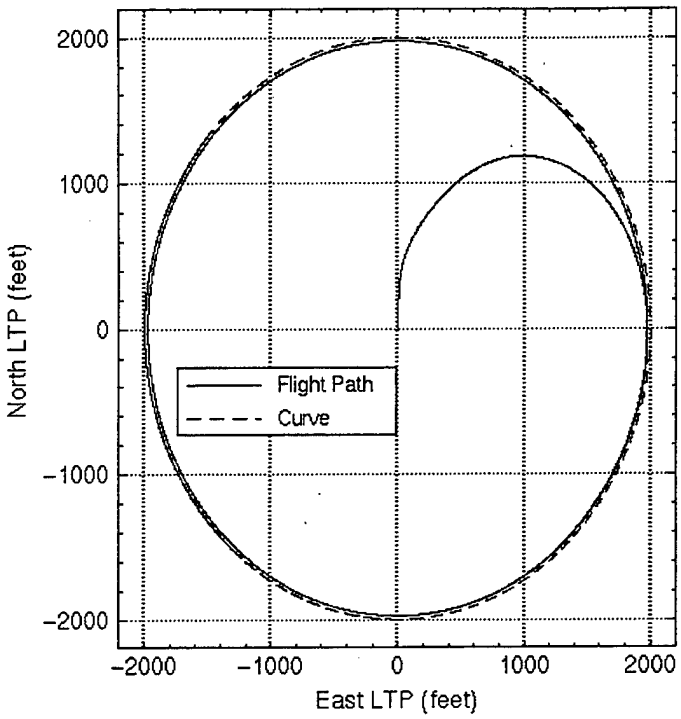

Figure 44 Inside Circle

In both cases the controller was able to capture and track the desired course. Under these conditions the steady state tracking error was 83 feet on the inside of the circle. This error is due to the visibility distance of the control point being 1000 feet ahead of the closest point on the circle. As the aircraft moves forward, the control point continually moves to the inside of the turn. Even with $u = 0$ and $du/dt = 0$, this produces a ground track to the inside during any steady turn, as seen in Figure 45.

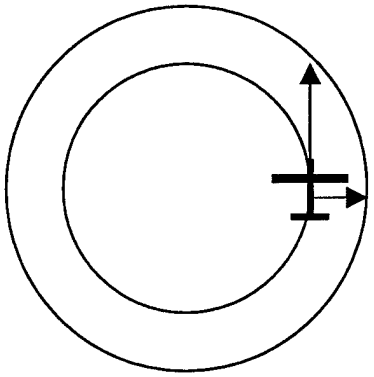

Figure 45 Steady State Error

The sine wave tested the controller over a more complex course with variable radius turns in both directions. Figure 46 shows the flight path with the aircraft initially on the desired track, and Figure 47 shows the results when starting off track. Both display good performance and demonstrate the robustness of the controller design.

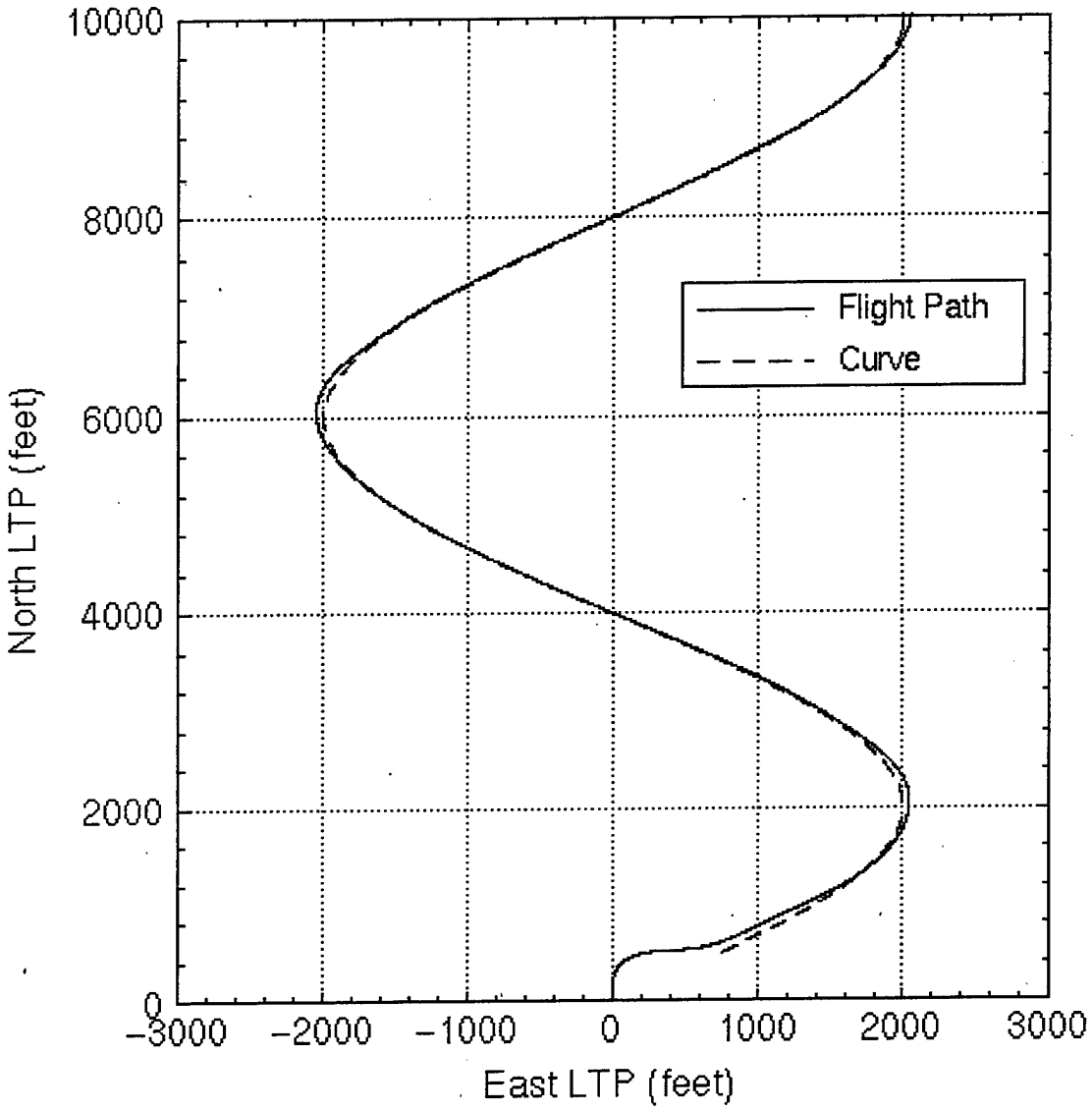

Figure 46 Sine Wave On Track

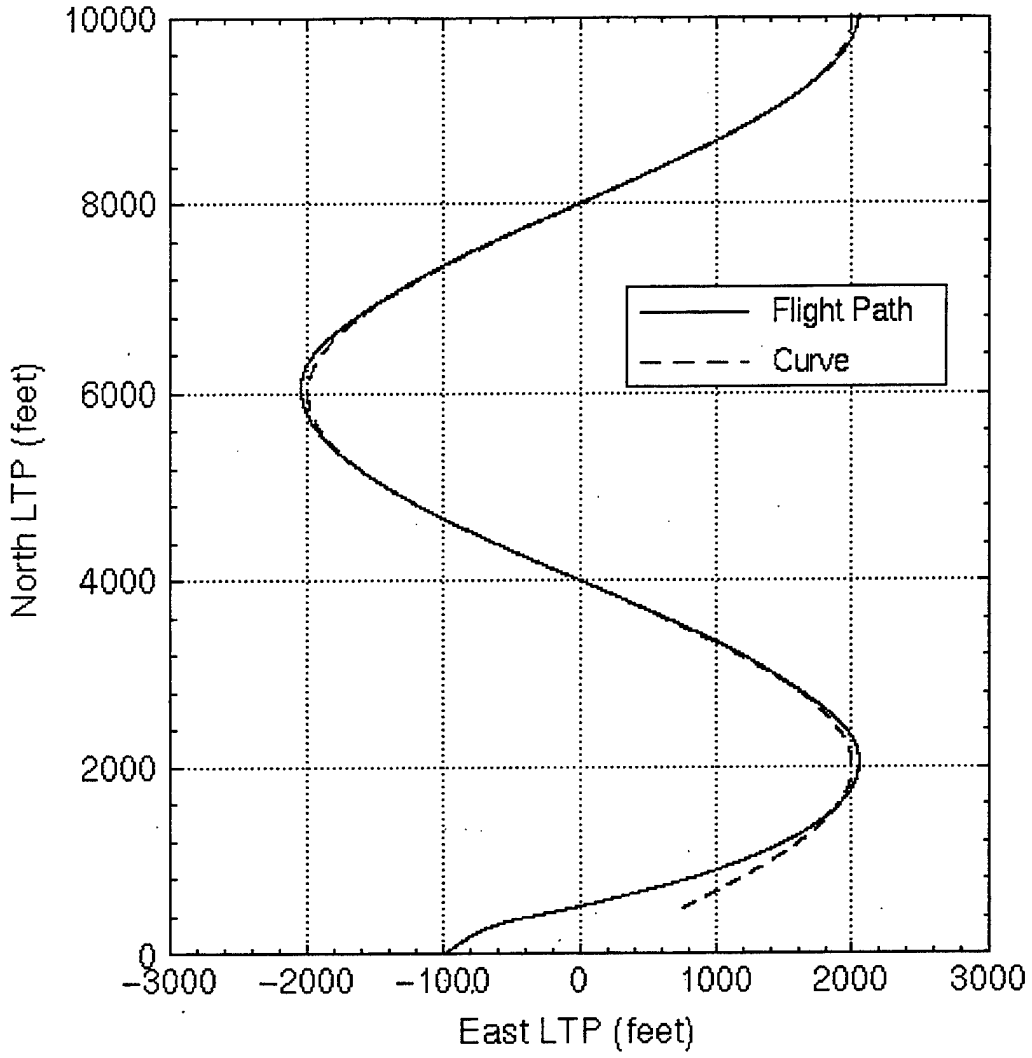

Figure 47 Sine Wave Off Track

3. Noise

Adding noise to u at the output of the sensor model further tested the robustness of the controller. Simulations with Guassian noise up to 50% of the maximum noise free value showed that the controller can track the desired curve. Figure 48 shows the camera output and tracking error for a noise level of 50%. The FROG does not currently have an accurate IMU, so simulations with ±30° of heading noise with a constant 5° bias were made

to ensure that the sensor model would perform during flight tests. Bank and pitch angles were also set to zero to test the effect of incorrect values from the IMU. Once the actual camera system is installed angle errors will not be critical, since they are only used in the sensor model.

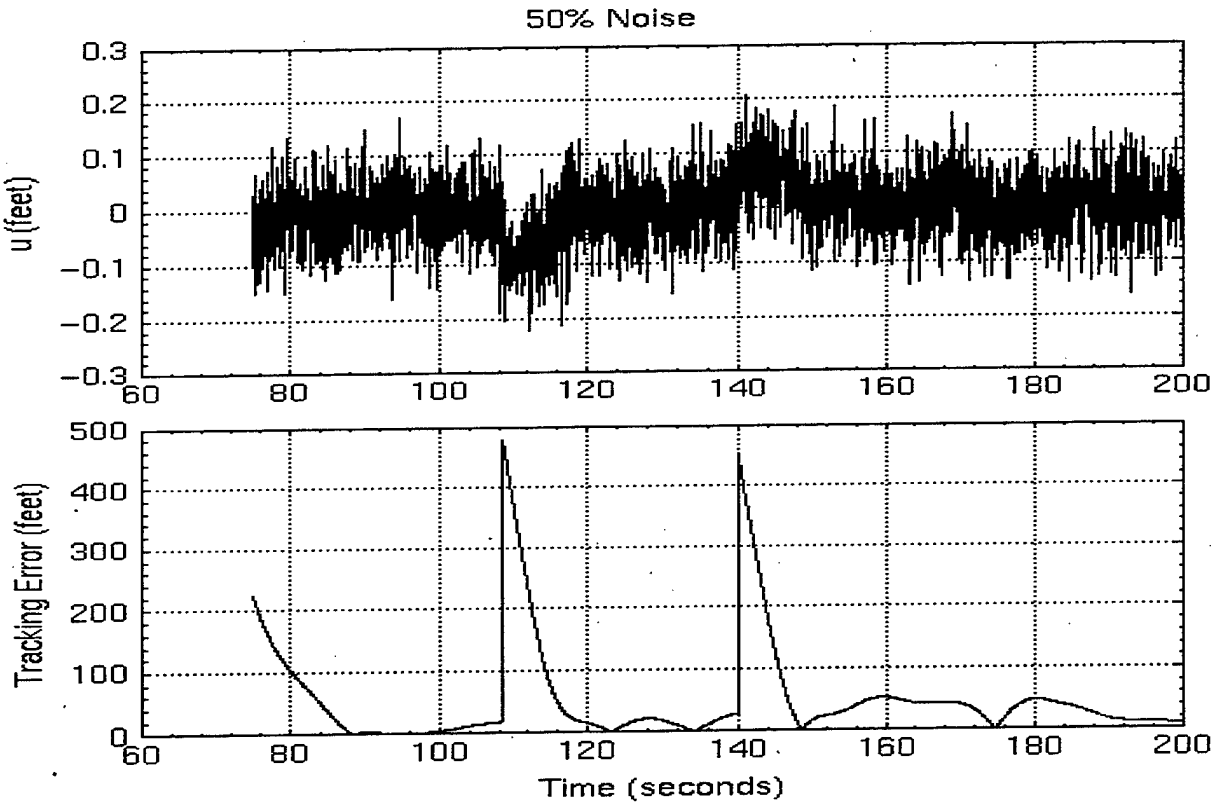

Figure 48 Camera Output/Tracking Error

4. Wind

Tests were done to investigate the wind effects on the controller. Figure 49 shows the flight path with a constant 10 knots of wind from the northeast. As can be seen, an offset downwind of the desired track was flown. Tests with the wind out of the north were also made and, as expected, showed an offset only on the crosswind leg.

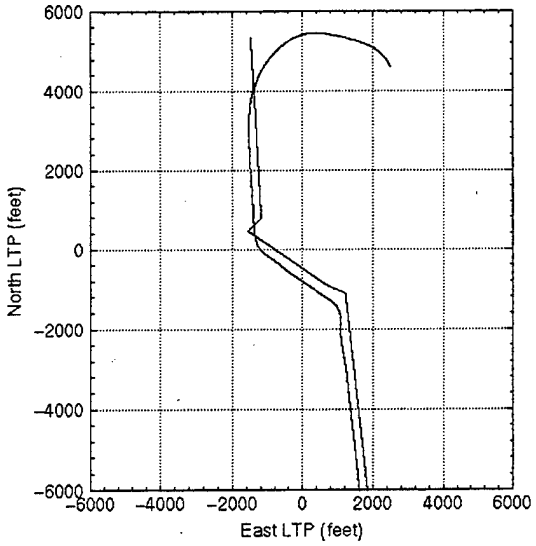

Figure 49 10 Knot NE Wind

This is a result of controlling the aircraft in {IM}. When the aircraft has established a constant ground track with zero lateral displacement in {IM}, a steady state tracking error is experienced as seen in Figure 50. This error is a function of the wind crab angle and the visibility distanced used for the control point.

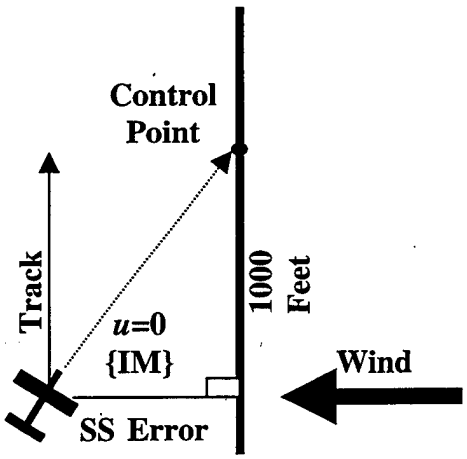

Figure 50 Wind Induced Error

B. FLIGHT TESTS

1. Flight One

Two flight tests were made to test the controller design and to validate the use of the computer based camera model in the place of an actual camera. During flight tests the camera model uses position from the GPS and Euler angles from the IMU exactly as those values from the FROG model superblock were used during ground simulations. The flights were conducted at the Chualar RC airfield shown in Figure 19.

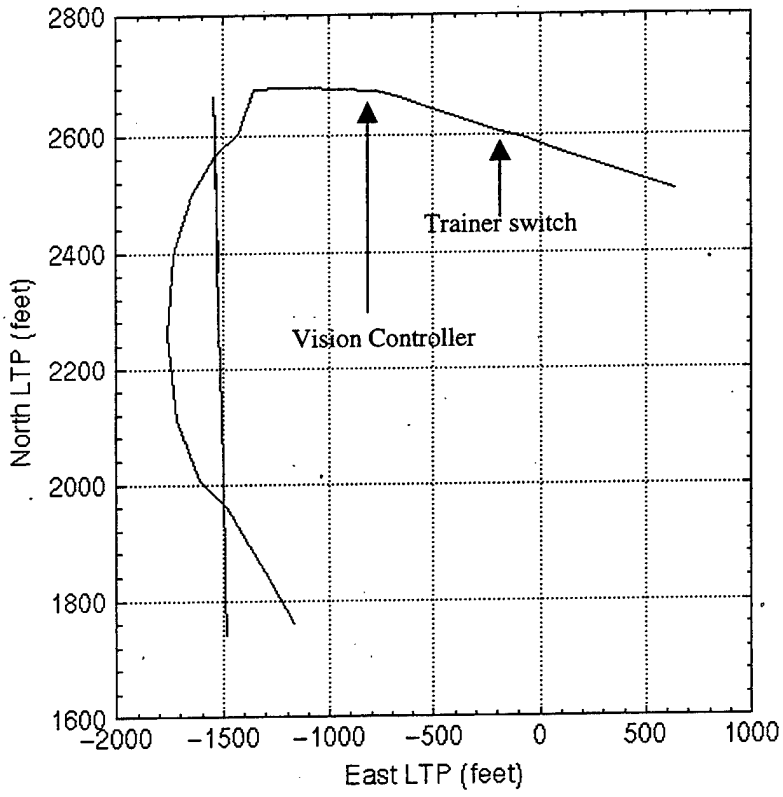

Figure 51 Initial Capture Flight Path

Figure 51 shows the initial capture and track results of the first run. After takeoff the FROG was manually flown by the pilot to a point north of the field and established on a westerly heading to intercept the simulated river before control is passed to the computer.

At five seconds the trainer switch was engaged by the pilot handing over control, as evidenced by the yaw rate commanded shown in Figure 52. The heading controller then commanded a 230 heading towards the river. From the GPS tracking data shown in Figure 51 it can be seen that the FROG was actually on a northwesterly heading. This is due to the poor performance of the onboard IMU. At ten seconds the range to the line had decreased to 700 feet, which changed states in the capture/track state machine. Control was then switched to the vision controller. At the switching point the output of the sensor model shows the Image plane coordinate u of the control point to the left of the aircraft. A negative yaw rate commanded was sent, as expected. As the FROG entered a left turn, u become more negative. In other words, it was going farther to the left. This is a good display of the adverse yaw characteristic of the FROG. As the control point passed in front of the FROG the yaw rate commanded went from negative to positive exhibiting the proper operation.

The initial capture attempt shown in Figures 51 and 52 demonstrates that the sensor model is working, that the capture/track state machine has the proper logic, and that the controller is sending reasonable commands. However, the FROG was over controlled and unable to track the line. The run was terminated before divergence for safety reasons. This result can be attributed to two factors. The first reason is the smaller stability margins with the controller gain set at 75. The second reason is the large cross track angle when the vision controller was engaged. This last problem was due to both the difficulty in manually positioning the FROG with relation to the line from the ground and to the poor heading from the IMU. Further tests runs were made with similar results.

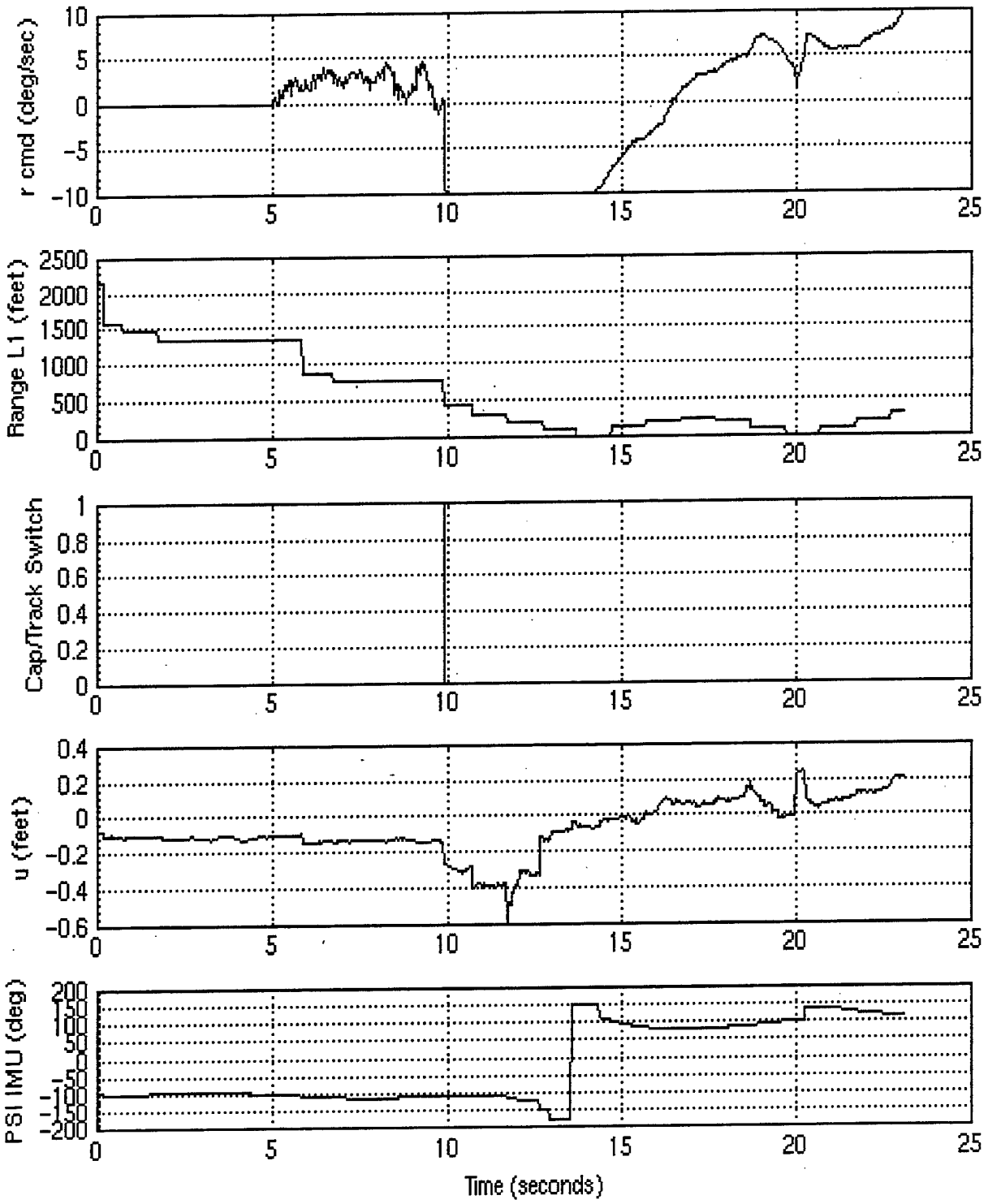

Figure 52 Initial Capture Data

2. Flight Two

A second flight was conducted after insertion of a sliding gain in the SystemBuild model of the controller. The initial gain was set at 50 to increase the stability margins as discussed in Chapter V. Figure 53 shows the track of the FROG with this setting. Control was passed to the computer with the aircraft closer to the airfield to aid in positioning the FROG visually by the pilot. The trainer switch was engaged near point two after the sensor model has switched to line two.

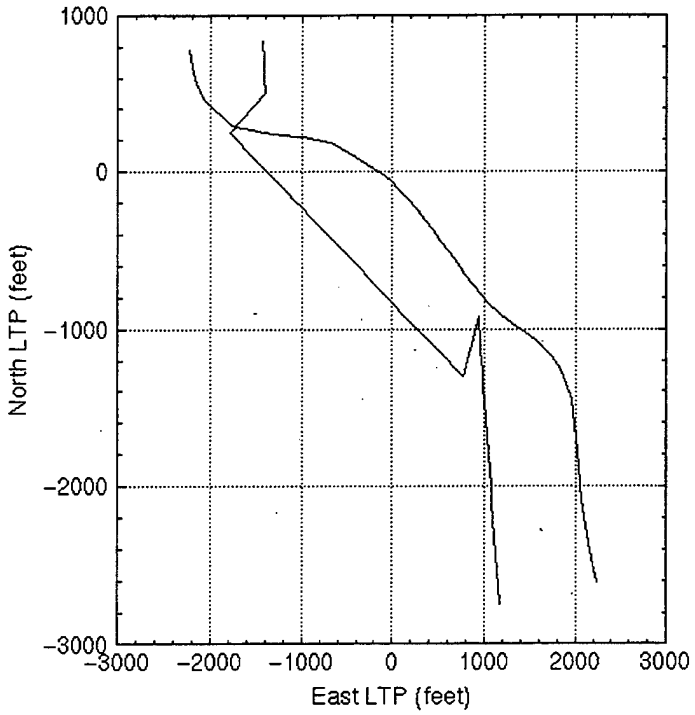

Figure 53 Gain 50 Flight Path

The sensor output, *u*, seen in Figure 54 shows that the line was to the right of the FROG and that a positive yaw rate was properly commanded to track the line. Starting at 20 seconds the commanded yaw rate has caused the FROG to begin turning back towards

the line. However, by this time the range to point three has decreased to 600 feet, and the line point state machine switches to line three. At 38 seconds the adverse yaw characteristic of the FROG again caused a large spike in *u* and the commanded yaw rate. The controller handled this spike with the rate limiter and continued to command a positive yaw rate to regain the new track. The test run ended before a steady state track occurred due to the constraints of the flight test area. Additional runs were made under similar conditions duplicating the results.

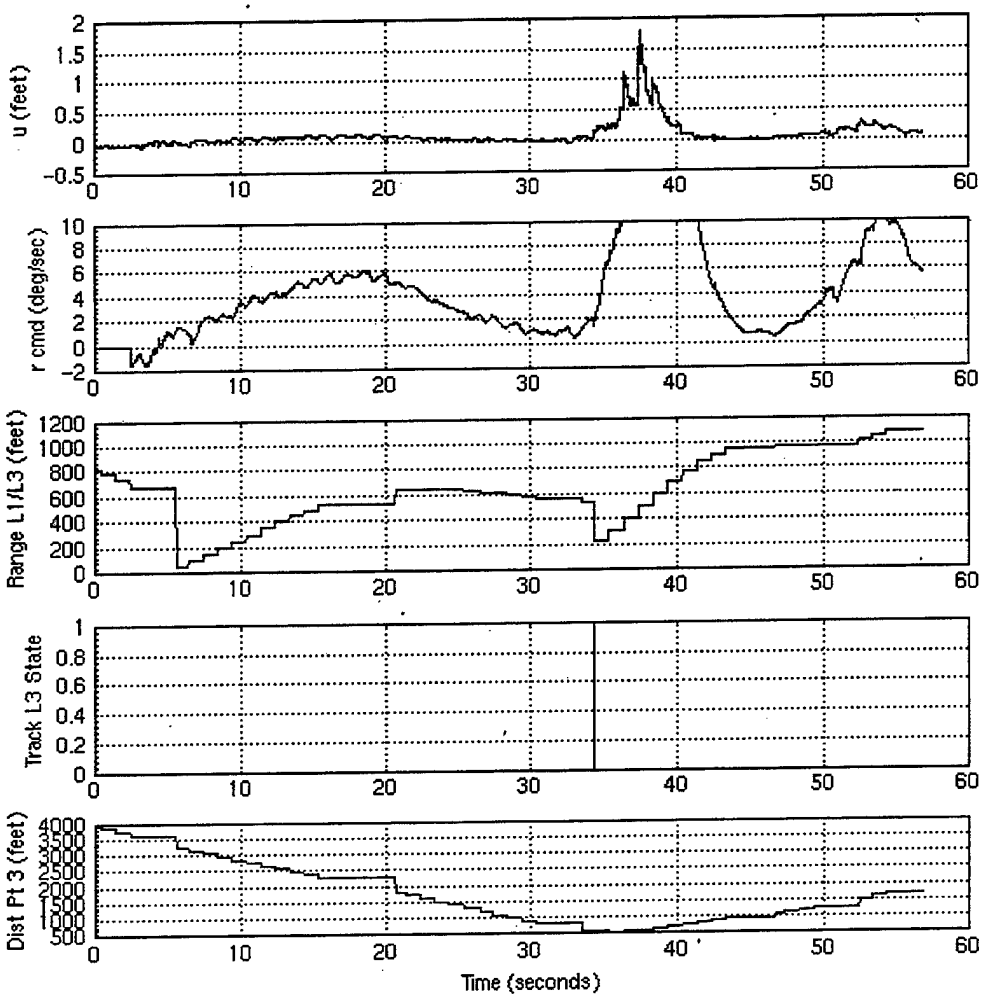

Figure 54 Gain 50 Data

VII. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The objectives of this thesis as stated in the introduction were accomplished. A vision controller for the FROG UAV was developed and successfully tested both in the lab and in actual flight. An accurate model for the sensor package was designed and used throughout the process.

Poor IMU performance and the difficulty encountered in visually positioning the FROG prevented the flight testing of the system over the full three line course. It also led to the reduction of the gain to a level that the controller was not able to perform in the limited flight test area. If the initial intercept geometry could be achieved, then the higher gain may be closer to the proper setting.

The decision to work in the Image plane led to a steady state tracking error with any crosswind component present. Taking the output of the sensor and transforming it to an Inertial reference frame is an option for eliminating this error, if accurate onboard sensors are available. However, this may not be desired. The goal of a vision guided vehicle is to keep the path being followed in the field of view. Slow vehicles, such as UAVs, require large crab angles to compensate for wind to follow a given track over the ground. When the UAV is exactly above the desired track, a strong crosswind could produce a crab angle large enough that the track would be out of the field of view of the sensor. A decision must be made based on the mission and the sensor onboard. Of course, a movable sensor would

eliminate this concern, but it would necessitate the measurement of variable rotation angles to transform from {B} to {C}.

The secondary goal of improving the Flight Management System was also accomplished. These improvements will enhance the safety and operational effectiveness of the existing controllers during flight testing of the FROG.

B. RECOMMENDATIONS

Since only two test flights were made, it is recommended to continue flight testing and refining the current vision controller and the sensor model. Flights along a single line could be used to fully test the stability of the controller in flight using variable gains. Adjusting the visibility distance should also be investigated. Ultimately a more complex controller may be needed.

A more accurate heading source is needed to help with the initial capture and track problem. Changing the state machine logic that switches from the heading to the vision controller to include a heading check should be tested. This would ensure that the heading is within the stable intercept cone before switching.

Additional flights along the same course in a northerly direction should also be made. Other courses should be designed that remain within the confines of the flight test area. When the software design and hardware interfaces for the actual camera and digital image processor are complete, they should be flight tested.

A controller that uses the transformation of the camera output to an Inertial reference frame could be developed for laboratory testing. Actual flight testing, however, would not be productive with the existing IMU.

APPENDIX CONTROL POINT CALCULATIONS

This Xmath code is used in straight line simulations by the user defined block titled" calc pt LTP" as described in Chapter IV. Tests to prevent division by zero are also made. Default values are supplied for these cases.

```
inputs:  u;
outputs:  y;

float u(7), y(6), num, dem, m,Xc,Yc,Xa,Ya,direction, d;

d = 1000;

num = u(6) - u(4);
dem = u(5) - u(3);

if abs(dem) < 0.001 then
  dem = 0.001*sign(dem);
endif;

if dem == 0.0 then
  dem = 0.001;
endif;

if abs(num) < 0.01 then
  num = 0.01*sign(num);
endif;

if num == 0.0 then
  num = 0.01;
endif;

m = num/dem;

Yc = (u(2)*m^2 + u(1)*m - u(5) *m + u(6))/(1 + m^2);
Xc = (Yc - u(6))/m + u(5);

if u(7) > 0.5 then
```

```
  direction = 1.0;
else
  direction = -1.0;
endif;

Xa = Xc + d*direction*cos(atan(m));
Ya = m*(Xa - Xc) + Yc;

y(1) = ((Xc - u(1))^2 + (Yc - u(2))^2)^0.5;
y(2) = Xc;
y(3) = Yc;
y(4) = Xa;
y(5) = Ya;
y(6) = 0.0;
```

This is the Xmath code used to simulate a circle. It uses the same methodology as the line calculations.

```
inputs:  u;
outputs:  y;

float u(2), y(6), num, dem,Xc,Yc,Xa,Ya, r, d, theta, dtheta;

r = 2000;
d = 1000;

num = u(1);
dem = u(2);

if abs(dem) < 0.001 then
  dem = 0.0001*sign(dem);
endif;

if dem == 0.0 then
  dem = 0.0001;
endif;

if abs(num) < 0.001 then
  num = 0.001*sign(num);
endif;

if num == 0.0 then
```

```
  num = 0.001;
endif;

theta = atan2(num,dem);
dtheta = d/r;

Xc = r*cos(theta);
Yc = r*sin(theta);

Xa = r*cos(theta + dtheta);
Ya = r*sin(theta + dtheta);

y(1) = ((Xc - u(1))^2 + (Yc - u(2))^2)^0.5;
y(2) = Xc;
y(3) = Yc;
y(4) = Xa;
y(5) = Ya;
y(6) = 0.0;
```

This is the Xmath code for the sine wave simulations.

```
inputs: u;
outputs: y;

float u(2), y(6), num, dem,Xc,Yc,Xa,Ya, r, d, p;

d = 500;
r = 2000;
p = 8000;

Xa = u(1) + 500;
Ya = r*sin(2*3.1415*Xa/p);

Xc = Xa;
Yc = Ya;

y(1) = ((Xc - u(1))^2 + (Yc - u(2))^2)^0.5;
y(2) = Xc;
y(3) = Yc;
y(4) = Xa;
y(5) = Ya;
y(6) = 0.0;
```

LIST OF REFERENCES

1. Froncillo, S.J., *Design of Digital Control Algorithms for Unmanned Air Vehicles*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1998.

2. Komlosy, J.A., *Applications of Rapid Prototyping to the Design and Testing of UAV Flight Control Systems*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1998.

3. Rivers, T.C., *Design and Integration of a Flight Management System for the Unmanned Air Vehicle FROG,* Engineer's Thesis, Naval Postgraduate School, Monterey, CA, December 1998.

4. Kaminer, I.I., *AA3276: Intro to Avionics*, Course Notes, Naval Postgraduate School, Monterey, CA, September 1996.

5. Ogata, K., *Modern Control Engineering*, chapt. 7, Prentice-Hall,1997.

6. Papageorgiou, E. C., *Development of a Dynamic Model for a UAV*, Masters Thesis, Naval Postgraduate School, Monterey, CA, March 1997.

INITIAL DISTRIBUTION LIST

No. of copies

1. Defense Technical Information Center.....2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, Virginia 22060-6218

2. Dudley Knox Library.....2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, California 93943-5101

3. Doctor Isaac I. Kaminer, Code AA/KA.....3
   Department of Aeronautics and Astronautics
   Naval Postgraduate School
   Monterey, California 93943-5121

4. Doctor Russell Duren, Code AA/DR.....1
   Department of Aeronautics and Astronautics
   Naval Postgraduate School
   Monterey, California 93943-5121

5. Doctor Conrad F. Newberry, Code AA/NE.....1
   Department of Aeronautics and Astronautics
   Naval Postgraduate School
   Monterey, California 93943-5121

6. Department of Aeronautics and Astronautics.....1
   Code AA
   Naval Postgraduate School
   699 Dyer Rd. Rm. 137
   Monterey, California 93943-5106

7. Lieutenant Commander Mark T. Watson.....3
   28 Bridgetown Bend
   Coronado, California 92118